Fysiologisia suureita mittaavan hajautetun järjestelmän suunnittelu ja toteutus

Juha-Matti Santero

Sähkötekniikan korkeakoulu

Diplomityö, joka on jätetty opinnäytteenä tarkastettavaksi diplomi-insinöörin tutkintoa varten Espoossa 23.11.2015.

Työn valvoja:

Prof. Raimo Sepponen

Työn ohjaaja:

TkL Matti Linnavuo

Aalto-yliopisto
Sähkötekniikan korkeakoulu

AALTO-YLIOPISTO
SÄHKÖTEKNIIKAN KORKEAKOULU

DIPLOMITYÖN
TIIVISTELMÄ

Tekijä: Juha-Matti Santero

Työn nimi: Fysiologisia suureita mittaavan hajautetun järjestelmän suunnittelu ja toteutus

Päivämäärä: 23.11.2015 Kieli: Suomi Sivumäärä: 7+53

Sähkötekniikan ja automaation laitos

Professuuri: Sovellettu elektroniikka Professuurikoodi: S-66

Työn valvoja: Prof. Raimo Sepponen

Työn ohjaaja: TkL Matti Linnavuo

Tämän diplomityön tavoite oli aloittaa uuden järjestelmän kehitystyö, jota voidaan käyttää terveydentilan seurantaan. Alustavasti Terveystuoli-järjestelmäksi nimetty järjestelmä suunniteltiin ja toteutettiin. Idean lähteenä oli Aalto-yliopistossa aikaisemmin kehitetty Terttu-terveystuoli, josta annettua palautetta käytettiin uuden järjestelmän teknisessä määrittelyssä. Parannuksiksi Terttu-terveystuoliin nähden tavoiteltiin yhdistettävyyttä ulkoiseen tiedonhallintajärjestelmään, käytettävyyttä mobiililaitteelta ja järjestelmän kokonaiskustannusten vähentämistä.

Tässä diplomityössä määriteltiin uuden järjestelmän toiminnot, valittiin toimintoihin sopiva laitteisto ja esiteltiin laitteiston käyttämiseksi vaadittua ohjelmistoa. Terveystuoli-järjestelmä jaettiin kolmeen fyysisesti erilliseen osajärjestelmään: mittausjärjestelmään, verkkosovellukseen ja palvelinsovellukseen. Lopuksi järjestelmä toteutettiin prototyyppinä ja sen toimivuus tarkastettiin. Se sisälsi toimivat sydänsähkökäyrä-, verenpaine- ja veren happisaturaatiomittaustoiminnot. Sitä voitiin käyttää verkkoyhteyden kautta useilla eri tietokoneilla ja mobiililaitteilla. Järjestelmän jatkokehitystä suositellaan.

Avainsanat: Hajautetut järjestelmät, teollinen internet, terveystuoli, fysiologiset mittaukset

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Juha-Matti Santero

Title: Design and implementation of a distributed system for measurement of physiological quantities

Date: 23.11.2015 Language: Finnish Number of pages: 7+53

Department of Electrical Engineering and Automation

Professorship: Applied Electronics Professorship Code: S-66

Supervisor: Prof. Raimo Sepponen

Advisor: Lic.Sc. (Tech.) Matti Linnavuo

The aim of this thesis was to start the development of a new system for patient health monitoring. The system was designed and implemented. The idea for the system was based on a health chair previously developed in Aalto-university. Feedback from the health chair was used in technical definition of the new system. Enabling connectivity to an external information management system, use of mobile device as a user interface, and reduction of overall costs were pursued as the main improvements.

The system's features were defined, suitable hardware was chosen, and necessary software for using the hardware was described. The system was divided into three physically separate subsystems: measurement system, web application and server application. Finally the system was implemented as a prototype and its usability was checked. The prototype included working functionality for the measurement of electrocardiogram, blood pressure, and blood oxygen saturation. Prototype could be used via several computers and mobile devices. Further development of the system was recommended.

Keywords: Distributed systems, internet of things, health chair, physiological measurements

Esipuhe

Haluan kiittää Professori Raimo Sepposta ja ohjaajaani Matti Linnavuota diplomityöni mahdollistamisesta, ohjauksesta sekä hyvistä neuvoista. Kiitokset Heikki Ruotoistenmäelle mekaanisen toteutuksen avustamisesta ja Borys Plyenkoville Node.js:n opettamisesta. Tämän diplomityön kokeellisessa osuudessa käytettiin Aalto-yliopiston Health Factoryn tarjoamia tiloja ja laitteita.

Otaniemi, 19.10.2015

Juha-Matti Santero

Sisällysluettelo

Tiivistelmä ii

Tiivistelmä (englanniksi) iii

Esipuhe iv

Sisällysluettelo v

Lyhenteet vii

1 Johdanto 1

2 Fysiologiset mittaukset 2
2.1 Elektrokardiogrammi . . . 2
2.2 Pulssioksimetria . . . 4
2.3 Verenpaine . . . 5
2.4 Lämpötila . . . 5
2.5 Veren glukoosipitoisuus . . . 6
2.6 Paino . . . 6
2.7 Bioimpedanssi . . . 7

3 Terttu-terveystuolin arviointi ja uuden järjestelmän ideointi 9

4 Terveystuoli-järjestelmän suunnittelu 11
4.1 Mittausjärjestelmä . . . 11
4.1.1 Mittauslaitteisto . . . 12
4.1.2 Tietokoneen valinta . . . 16
4.1.3 Mittausjärjestelmän toiminnallinen määrittely . . . 19
4.2 Palvelinsovellus . . . 19
4.2.1 Internet-protokollan käyttö palvelinsovelluksessa . . . 20
4.2.2 Ohjelmointikielen ja ohjelmistoalustan valinta . . . 21
4.2.3 Palvelinsovelluksen ylläpito . . . 22
4.2.4 Palvelinsovelluksen toiminnallinen määrittely . . . 23
4.3 Käyttölaite . . . 23
4.3.1 Käyttöliittymässä vaaditut toiminnot . . . 23
4.3.2 Tiedonsiirto verkkosovelluksen ja palvelinsovelluksen välillä . . . 24
4.3.3 HTML 5 WebSocket-protokolla . . . 25
4.3.4 Verkkosovelluksen toiminnallinen määrittely . . . 25

5 Terveystuoli-järjestelmän prototyypin toteutus 27
5.1 Mittausjärjestelmän toteutus . . . 27
5.1.1 Arduino Yunin 32U4-mikrokontrollerin käyttöönotto ja konfigurointi . . . 27
5.1.2 Linux-ympäristön konfigurointi . . . 28
5.1.3 Python-skripti . . . 29

5.1.4 Verkkoyhteys mittausjärjestelmän ja palvelinsovelluksen välillä 33
5.1.5 Mittausjärjestelmän runko 33
5.2 Palvelinsovelluksen toteutus 34
5.2.1 Yhteys mittausjärjestelmään 35
5.2.2 Yhteys verkkosovellukseen 36
5.2.3 Verkkoyhteyksien hallinta 37
5.3 Verkkosovelluksen toteutus 37
5.3.1 Ohjelmointi 38
5.3.2 EKG-käyrän piirtäminen 39
5.3.3 Käyttöliittymän muodostaminen HTML-elementeillä 40
5.4 Prototyypin testaus 41
5.4.1 Metodologia 41
5.4.2 Tulokset 42
5.4.3 Tulosten tulkinta 46

6 Johtopäätökset ja suositukset **48**
6.1 Johtopäätökset 48
6.2 Suositukset 48

Viitteet **50**

Lyhenteet

BIA	bioelectrical impedance analysis
EKG	elektrokardiogrammi
Hb	hemoglobin
HbO2	oxyhemoglobin
HTML	hypertext markup language
HTTP	hypertext transfer protocol
HTTPS	hypertext transfer protocol secure
IDE	integrated development environment
JSON	JavaScript Object Notation
NIBP	noninvasive blood pressure
OEM	original equipment manufacturer
PTT	pulse transit time
SCP	secure copy protocol
SPI	serial peripheral interface
SpO2	peripheral capillary oxygen saturation
SSH	Secure Shell
TCP/IP	transmission control protocol/Internet protocol
TTL	transistor-transistor logic
UDP	user datagram protocol
USB	universal series bus
VOIP	voice over IP
WLAN	wireless local area network

1 Johdanto

Terttu-terveystuoli on Aalto-yliopiston Health Factory -projektin kehittämä mittausjärjestelmä. Se on tarkoitettu fysiologisten parametrien epäinvasiiviseen ja käytännölliseen mittaamiseen. Tertun suunniteltu käyttöympäristö on ensisijaisesti sairaaloissa, klinikoilla ja hoitolaitoksissa. Pääasiallisiin käyttäjiin kuuluvat vanhusten ja muiden huonokuntoisten potilaiden parissa työskentelevät hoitajat, joiden työhön kuuluu potilaiden terveydentilan pitkäaikaisseuranta. Terttu-terveystuoli on suunniteltu mittaamaan henkilön painoa, verenpainetta, sykettä, veren happisaturaatiota ja kehon bioimpedanssia. Nämä mittaukset ovat yleisiä toimenpiteitä terveydenhoitolaitoksissa, mutta niitä ei yleensä pystytä mittaamaan samanaikaisesti. Terttu-terveystuolin erityisominaisuus on useiden mittausten yhdistäminen yhteen järjestelmään, jolloin useita mittauksia suorittaessa ei ole tarpeen käyttää eri laitteita tai siirtää potilasta mittauksien välillä. Tämä tekee mittaamisesta nopeampaa ja vaivattomampaa hoitohenkilökunnalle ja potilaille. Mittaustapahtuma voi olla fyysisesti raskasta varsinkin huonokuntoisille potilaille, joiden kohdalla Terttu-terveystuolin hyödyt korostuvat. [1]

Tämän diplomityön tavoite on aloittaa uuden järjestelmän suunnittelu, jota voidaan käyttää terveydentilaan liittyvien suureiden mittaamiseen ja mittaustulosten esittämiseen. Järjestelmä on nimetty Terveystuoli-järjestelmäksi, ja se on Aalto-yliopiston Health Factoryn kehittämään Terttu-terveystuoliin perustuva innovaatio. Tässä diplomityössä tarkastellaan Terttu-terveystuolin toimintoja, sekä pohditaan miten ne voitaisiin toteuttaa paremmin. Kehitysideoiden pohjalta suunnitellaan Terveystuoli-järjestelmän arkkitehtuuri, joka perustuu käytettävissä olevaan laitteistoon. Arkkitehtuurin suunnittelussa tutkitaan laitteiston integrointiin tarvittavia liitäntöjä, verkkoyhteyksiä ja tietokoneohjelmia. Avoimen lähdekoodin ratkaisuja suositaan, jotta niitä voitaisiin mukauttaa mahdollisimman paljon. Lopuksi Terveystuoli-järjestelmä toteutetaan korkean tarkkuuden prototyyppinä. Prototyypin tavoite on varmistaa arkkitehtuurin toimivuus ja olla käytettävissä Terveystuoli-järjestelmän jatkokehityksessä.

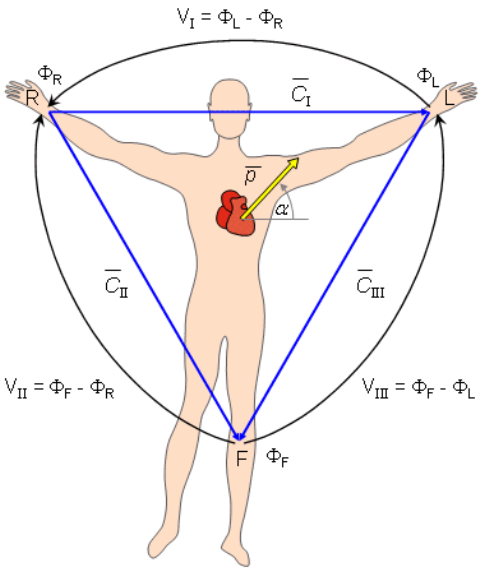
Luku 2 on johdanto fysiologisiin mittauksiin ja luku 3 sisältää Terttu-terveystuolin esittelyn ja arvioinnin. Luvussa 4 Terveystuoli-järjestelmä jaetaan erillisiin osiin, joilla on omat tehtävänsä. Laitteisto ja laitteiston valintaperusteet esitellään. Laitteiston käyttöön liittyviä tekniikoita ja menetelmiä esitellään. Lukuun 5 sisältyy Terveystuoli-järjestelmän toteutus toiminnallisena prototyyppinä. Prototyypin yleinen toimivuus tarkastetaan. Ehdotuksia Terveystuoli-järjestelmän kehittämiseksi esitetään. Luvussa 6.1 esitellään tämän diplomityön johtopäätökset ja suositukset kehitysprojektin edistämiseksi.

2 Fysiologiset mittaukset

2.1 Elektrokardiogrammi

Elektrokardiogrammi (EKG) kuvaa sydämen aiheuttamia sähköisiä potentiaaleja, joita voidaan havaita eri puolilla kehoa. EKG:ta mittaamalla voidaan saada tietoa sydämen sähköisestä toiminnasta ja mahdollisista toimintahäiriöistä. Esimerkiksi harvalyöntisyys, tiheälyöntisyys, rytmihäiriö ja eteis- tai kammiovärinä voidaan selvittää. EKG-mittauksen suorittamiselle ja tulosten tulkinnalle on luotu standardimalli, jotta eri mittaukset olisivat keskenään verrannollisia. [3]

EKG mitataan käyttämällä kytkentöjä, jotka mittaavat sähköistä potentiaalieroa kahden eri pisteen välillä. Sydämen aiheuttaman sähköisen potentiaalin lähde mallinnetaan usein dipolilähteeksi, joka sijaitsee keskellä kehoa. Einthovenin kehittämässä EKG-mallissa mittauspisteet sijaitsevat yhtä kaukana toisistaan, muodostaen tasakylkisen kolmion. Kytkennän mittaama jännite on verrannollinen dipolilähteen sähkökenttävektorin ja kolmion kyljen projektioon. Kolmiota kutsutaan Einthovenin kolmioksi sen keksijän mukaan. Kytkennät ovat havainnollistettu kuvassa 1. Kolmen mittauspisteen välille voidaan muodostaa kolme kytkentää, jotka selittävät yli 90 % sydämen sähköisestä toiminnasta. Rajoituksena on, että mittaus rajoittuu raajojen muodostamalle tasolle (frontaalitasolle), jolloin sähkökenttävektorin syvyyskomponentteja ei voida mitata. Tavanomainen 12-kytkentäinen EKG-mittaus tuottaa tarkimman tuloksen ja suurimman kliinisen hyödyn. 12-kytkentäinen mittaus toteutetaan käyttämällä raajaelektrodien lisäksi rinnalle ja kylkiin kiinnitettäviä elektrodeja, jolloin sähkökenttävektorin syvyyskomponentitkin voidaan selvittää. [3]



Kuva 1: EKG-raajakytkennät ja sydämen mallinnus dipolilähteenä. [3] Kuvaa muokattu

Kytkennät vastaavat sähköisen potentiaalin eroa kahden pisteen välillä. Einthove-

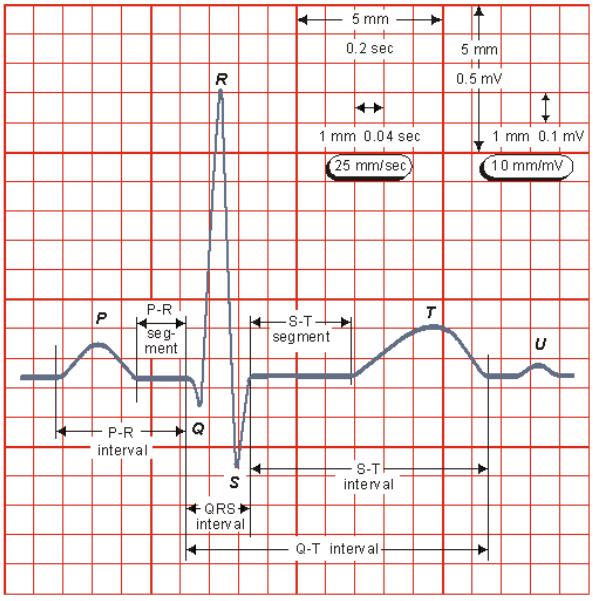
nin menetelmässä raajakytkennät ovat määritelty seuraavasti:

$$V_{\mathrm{I}} = \Phi_{\mathrm{L}} - \Phi_{\mathrm{R}}$$
$$V_{\mathrm{II}} = \Phi_{\mathrm{F}} - \Phi_{\mathrm{R}}$$
$$V_{\mathrm{III}} = \Phi_{\mathrm{F}} - \Phi_{\mathrm{L}}$$

jossa

V_{I} on I-kytkennän jännite
V_{II} on II-kytkennän jännite
V_{III} on III-kytkennän jännite
Φ_{L} on vasemman käden potentiaali
Φ_{R} on oikean käden potentiaali
Φ_{F} on vasemman jalan potentiaali [3]

EKG:n tulkinta perustuu käänteisen ongelman ratkaisuun, jossa sydämen sisäinen toiminta pyritään selvittämään ulkoisista mittauksista. Käänteiselle ongelmalle ei ole olemassa yksikäsitteistä ratkaisua. Kliinisessä käytössä EKG:ta voidaan käyttää joidenkin sairauksien tarkkaan diagnosointiin, mutta useiden sairauksien arvioinnissa tarkkuus ei ole täydellinen. Epäilty sairaus usein todennetaan muilla menetelmillä. Koska EKG:n mittaustekniikka on standardoitu, terveiden potilaiden mittaustulokset ovat samaa muotoa. Kuvassa 2 on esitetty normaali elektrokardiogrammi. Diagnoosi tehdään vertaamalla potilaan tuloksia normaaliin elektrokardiogrammiin, jolloin lääkäri tai muu asiantuntija voi tunnistaa eri sairauksia elektrokardiogrammin epätavallisten muotojen perusteella. [3]

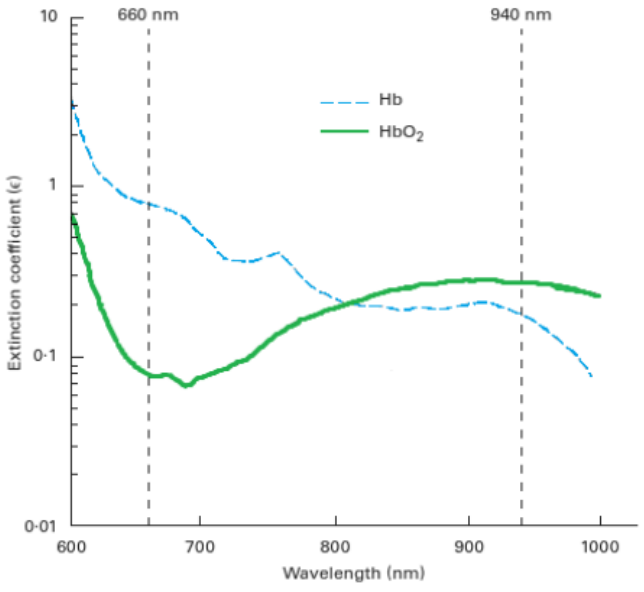


Kuva 2: Normaali elektrokardiogrammi ja sen osat. [3] Kuvaa muokattu

2.2 Pulssioksimetria

Pulssioksimetria on menetelmä, jolla mitataan potilaan veren happisaturaatiota (SpO2) ja sykettä ja jolla muodostetaan pletysmogrammi. Happisaturaatiolla tarkoitetaan kuinka suuri osa veren hemoglobiinista sillä hetkellä kuljettaa happea. Happisaturaatio ilmoitetaan hapellisen (HbO2) ja hapettoman hemoglobiinin (Hb) suhteesta prosenttilukuna, ja se on normaalisti noin 95 - 98 %. Tätä matalampi happisaturaatio voi olla merkki hengitysvaikeuksista ja korkeampi seurausta hyperventilaatiosta. Pletysmogrammi on kuvaaja tutkittavan ruumiinosan, yleensä korvanlehden tai sormenpään, tilavuuden muutoksesta ajan funktiona. Tilavuuden muutos aiheutuu sykkeestä, joka laajentaa verisuonia. Pulssioksimetri laskee sykkeen pletysmogrammista. Pletysmogrammilla varmistetaan myös, että happisaturaatiomittaus toimii luotettavasti. Sitä voidaan myös käyttää verenkierron ja sydäntoimintojen tutkimisessa. [7]

Happisaturaatiota voidaan mitata epäinvasiivisesti optisilla menetelmillä. Hapellisella ja hapettomalla hemoglobiinilla on erilaiset absorptiospektrit, joten ne voidaan laskea mittaamalla kudoksen läpäisseen tai siitä heijastuneen valon intensiteetin. Kun mittaus otetaan korvanlehdestä tai sormesta niin silloin tutkitaan kudoksen läpäissyttä valoa. Lihas- ja aivokudosmittauksessa tutkitaan heijastunutta valoa. Pulssioksimetrissa käytetään kahta eri aallonpituutta, joilla mitataan hapellisen ja hapettoman hemoglobiinin aiheuttamaa absorptiota. Aallonpituudet ovat valittu niin, että niiden välillä on selvästi havaittava absorptioero. Hapellisen ja hapettoman hemoglobiinin suhde voidaan selvittää laskemalla aallonpituuksia vastaavien absorptioiden suhde. [7]



Kuva 3: Hapellisen (HbO2) ja hapettoman (Hb) hemoglobiinin absorptiospektrit ja pulssioksimetriassa käytetyn valon aallonpituudet. [7] Kuvaa muokattu

2.3 Verenpaine

Verenpainetta käytetään usein perustana potilaiden diagnosoinnille ja hoidolle. Verenpaine voidaan mitata katetrilla verisuonesta, mutta yleisemmin käytetään mansetilla mitattavia menetelmiä kuten auskultaatiota tai oskillometriaa. Molemmissa menetelmissä käytetään olkavarren ympäri kiristettävää mansettia. Kun mansetin ilmapussiin lisätään tarpeeksi ilmanpainetta, se puristaa olkavarren ympäriltä ja estää hetkellisesti verenkierron. Ilmanpainetta vähennetään kunnes verenkierto palautuu. Auskultaatiossa kuunnellaan verenkierron palautumisesta aiheutuvia ääniä, joita kutsutaan Korotkoff-ääniksi. Oskillometriassa mitataan mansetin paineen oskillaatiota verenkierron palautumisen aikana. Oskillaation amplitudi kertoo, milloin verenkierto on palautumassa puristuksesta. [8]

Verenpaineen suureina käytetään systolista ja diastolista verenpainetta. Ne kuuluvat yleisimmin mitattuihin fysiologisiin parametreihin, ja niiden kliininen merkittävyys on suuri. Korkea systolinen verenpaine muun muassa lisää riskiä sairastua sydän- ja verisuonitauteihin [10]. Systolinen verenpaine mitataan korkeimmasta mansetin paineesta, jolla verenkierto on mahdollista. Tätä painetta vastaa auskultaatiossa Korotkoff-äänien kuuluminen ja oskillometriassa paineoskillaation havaitseminen. Diastolinen verenpaine mitataan paineesta, jolla äänet ja oskillaatiot ovat kadonneet. Nykyaikaiset verenpainemittarit täyttävät mansetin, mittaavat sykkeen, systolisen ja diastolisen verenpaineen automaattisesti. [8] Verenpainemittaus ei ole jatkuva, joten se voidaan käynnistää vasta kun potilas on valmistautunut. Mansetin pukeminen ja riisuminen edellyttää vaivannäköä, varsinkin jos potilas on pukeutunut pitkähihaiseen vaatteeseen. Olkavarsi pitää riisua mansettia varten. Riippuen mansetin mallista, sen pukeminen sopivalle kireydelle voi olla hankalaa ilman apua. Liian löysä tai kireä kiinnitys voi vähentää mittauksen tarkkuutta [9].

2.4 Lämpötila

Kehon lämpötilan mittaus on laajalti käytetty menetelmä, jolla potilaasta voidaan havaita hypotermian tai tulehduksen merkkejä. Mittauksen tarkkuuteen vaikuttaa pääasiassa mittauslaitteen tarkkuus, mittauskohdan sopivuus ja mittaajan taito. Mittausvirhe voi johtaa tarpeettomaan tai viivästyneeseen hoitoon. Mittausvirheiden lisäksi kehon lämpötilassa on systemaattista ja satunnaista vaihtelua, jotka riippuvat muun muassa henkilön iästä sekä vuorokauden ajasta. Ihmisen ydinlämpötila voidaan mitata tarkimmin käyttämällä keuhkovaltimokatetria, mutta se on liian invasiivinen käytettäväksi päivittäisessä kliinisessä mittauksessa. Lämpötila mitataan useammin limakalvolta tai ihon pinnalta. Mittauskohtaa pidetään hyvänä, jos se on ilmaton ja sen läheltä kulkee suuri määrä valtimoverta. Limakalvolta mitattu lämpötila vastaa ydinlämpötilaa paremmin kuin ihon pinnalta mitattu, mutta se on huomattavasti epäkäytännöllisempää. Esimerkiksi mittaus suun tai muun ontelon kautta on häiritsevää, ja hygieniakysymykset lisäävät hoitajien työtä ja aiheuttavat ylimääräisiä kuluja. [11]

Lämpötilan epäinvasiivisessa mittauksessa on käytetty useita erilaisia mittauslaitteita. Kliinisessä ympäristössä mittauslaitteen tärkeitä ominaisuuksia ovat tarkkuus,

turvallisuus, nopeus ja käytön helppous. Elohopealämpömittarit ovat olleet suosittuja ja luotettavia mittalaitteita monien vuosien ajan, mutta elohopean ympäristö- ja terveyshaittojen takia niitä ei enää käytetä kliinisessä ympäristössä. Sähköiset lämpömittarit soveltuvat paremmin fysiologisiin mittauksiin, ja niiden käyttö on hyvin yleistä. Niiden noin 30 - 50 sekunnin vasteaika voi kuitenkin olla haitta joissain sovelluksissa. Infrapunavaloon perustuvat korvalämpömittarit ovat yleisesti käytettyjä vaihtoehtoja sähköisille lämpömittareille. Korvalämpömittari asetetaan korvakäytävään, jossa se mittaa lämpötilaa tärykalvon infrapunasäteilyn kautta. Mittaukseen voi aiheutua epätarkkuutta potilaalle ominaisista lähteistä, kuten korvakäytävän muodosta, korvavahasta ja korvakäytävän karvatupista. [11]

2.5 Veren glukoosipitoisuus

Veren glukoosipitoisuus, eli verensokeri, määritellään veren glukoosin määränä tilavuusyksikköä kohden. Elimistö säätelee verensokerin määrää pääasiassa insuliinin kautta. Normaalisti verensokeri muuttuu pitkin päivää, ja esimerkiksi ruualla, alkoholilla ja lääkkeillä on vaikutus verensokeriin. Jos verensokeri on normaalin alueen ulkopuolella, voi olla syytä epäillä sairautta. Hyperglykemia eli korkea verensokeri voi olla merkki diabeteksesta. Diabetes on krooninen sairaus, jota hoidetaan insuliinilla tai lääkkeillä, jotka tehostavat kehon oman insuliinin tuotantoa. Insuliinin sopivan annostelun määrittämiseksi voidaan seurata verensokerin määrää, ja siitä on hyötyä varsinkin tyypin 1 diabeteksen hoidossa. [12] Verensokeri voidaan mitata verestä koeliuskoilla ja verensokerimittarilla. Mittaus on invasiivinen ja edellyttää ihon puhkomista. Verensokeri on mitattava useasti, jotta diabeteksen aiheuttamat terveysriskit voidaan minimoida. [13]

Yleisessä käytössä olevat mittausmenetelmät ovat ongelmallisia, koska ne ovat invasiivisia. Invasiivista mittausmenetelmää on hankala automatisoida, koska se edellyttää ihon puhkomista, verinäytteen ottamista ja koeliuskan lukemista. Näiden ongelmien takia verensokerimittauksen kehitys on siirtynyt epäinvasiiviseen mittaukseen. Esimerkiksi absorptiospektroskopiaan ja Raman-spektroskopiaan perustuvia verensokerimittareita on jo markkinoilla. Kuitenkin niiden tarkkuus, robustisuus ja vakaus vaativat huomattavaa jatkokehitystä, eikä toistaiseksi saatavilla olevia laitteita ole tarkoitettu tavanomaisten mittausmenetelmien korvaajaksi. [13]

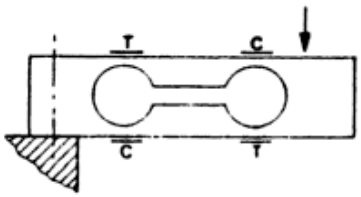
2.6 Paino

Painon mittaaminen on tärkeää liikalihavuuden hoidossa. Liiallinen kehonsisäisen rasvan määrä on yhdistetty sairauksiin kuten tyypin 2 diabetekseen, aivohalvaukseen, verenpainetautiin ja nivelrikkoon. [14] Kehonpainon ja painoindeksin hyödystä sairastumisriskin arvioinnissa on esitetty epäilyksiä [5][15], mutta käytäntö on silti yleinen. Vanhustenhoidossa painoa mitataan myös sopivien lääkeannosten määrittämiseksi ja ravitsemustilan seuraamiseksi. Vanhusten tapauksessa tärkeää on painon pysyminen vakaana [16].

Painon mittauksessa käytetään usein kuormakennoa, joka on yleinen nimitys useille erityyppisille voima-antureille. Yleisin anturityyppi kuormakennoissa on

venymäliuska-anturi, jota käytetään monilla kaupan ja teollisuuden aloilla. Anturin lisäksi kuormakennon tärkein osa jousielementti, jonka tarkoitus on muuttaa siihen kohdistuva kuormitus halutunlaiseksi venymäksi. Venymä on kohdistettava sellaisille kuormakennon alueille, joista se voidaan mitata tarkimmin. Jousielementin muodolla voidaan määrätä mitataanko taipumaa, suoraa jännitystä tai leikkausvoimaa. Ideaalisessa kuormakennossa venymä on suoraan verrannollinen siihen kohdistettuun kuormaan. Useat ilmiöt kuitenkin aiheuttavat epätarkkuutta voiman mittaamiseen kuormakennoilla. [17]

Ideaalisessa tapauksessa kuormakenno vastaa vain yhden akselin suuntaiseen voimaan ja on epäherkkä muista suunnista kohdistuville voimille. Sen lisäksi kuormakennon vasteen olisi oltava riippumaton siitä mihin kohtaan voima kohdistuu. Käytännössä täyttä riippumattomuutta ei voida saavuttaa, mutta kuormituskohdan vaikutus voidaan minimoida tietyn alueen sisällä. Kaksipalkkinen taipuva kuormakenno (ks. kuva 4) minimoi kuormituspisteen vaikutuksen rakenteensa ansioista. Taipuessa kyseisen kuormakennon kuormitettu pääty ei kallistu, vaan sen pinta pysyy samassa tasossa kuin kuormittamattomana. [17]



Kuva 4: Kaksipalkkinen jousielementti, joka minimoi kuormituspisteen paikan vaikutuksen mittaukseen. [17] Kuvaa muokattu

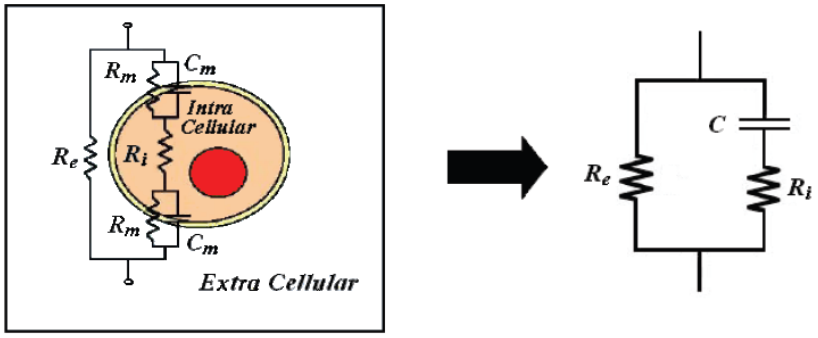
Painon tasainen jakautuminen kuormakennojen välille on tärkeää etteivät ne ylikuormitu. Kuormakennot kestävät yleensä 150 - 200 % ylikuormituksen ilman vaurioitumista. Ylikuormitus on kuormakennojen yleisin syy vikaantumiselle. Useaa kuormakennoa käytettäessä kuorman kokonaispaino jakautuu usean pisteen välille. Käytännössä kuorman tasapainotus on hankalaa, ja esimerkiksi neljän tukipisteen systeemissä suurin osa painosta on usein kolmen tai kahden tukipisteen varassa. Painon jakautumiseen voidaan vaikuttaa kiinnittämällä kuormakennot mahdollisimman tarkasti samalle tasolle. Epätasapainoa ei kuitenkaan voida poistaa kokonaan, koska viime kädessä sen määrittää lattian tai maanpinnan tasaisuus. [17]

2.7 Bioimpedanssi

Bioimpedanssianalyysi (BIA) on menetelmä, jolla voidaan tehdä johtopäätöksiä kehon koostumuksesta. Sitä voidaan käyttää aliravitsemuksesta kärsivien potilaiden hoidossa [4], mutta koostumus on myös yhdistetty yleiseen terveyteen. Millstein (2014, [5]) ehdottaa, että rasvan suhteellinen määrä kehossa voi olla mielekkäämpi suure useiden terveysriskien arvioinnissa kuin paino tai painoindeksi.

BIA perustuu kehon mallintamiseen sähköisenä ekvivalenttipiirinä, joka koostuu useista sarjaan- ja rinnankytketyistä komponenteista. Useita erilaisia ekvivalenttipii-

rejä on käytetty, ja niiden valinta riippuu tutkittavasta kohteesta ja tarkasteltavista ominaisuuksista. Yksi yleisimmistä malleista on Fricken ekvivalenttipiiri (ks. kuva 5), jossa solun välissä kulkevan virran lisäksi huomioidaan punasolujen kalvon ja solunesteen vaikutus. Vastus R_e kuvaa solunulkoisen nesteen resistiivisyyttä, R_i solunsisäistä resistiivisyyttä ja kondensaattori C solukalvojen kapasitanssia. [6] Solukalvon resistanssi R_m on erittäin suuri ja jätetään huomiotta [2].



Kuva 5: Fricken ekvivalenttipiirimalli. R_m on suuri ja voidaan jättää huomiotta. [2]

Fricken ekvivalenttipiirin impedanssia kuvaavasta kaavasta 1 voidaan nähdä, että impedanssi riippuu merkittävästi virran taajuudesta. Tällöin mittaukset voidaan kohdistaa solunsisäiseen tai solunulkoiseen nesteeseen käyttämällä vastaavasti korkeaa tai matalaa taajuutta. Tämä tarkoittaa myös, että nesteen kokonaismäärän mittaamiseksi on käytettävä useaa taajuutta. Nesteen mittaamisen lisäksi yksi- ja monitaajuus-BIA:ta on käytetty myös kehon rasvattoman massan arviointiin empiiristen mallien avulla. [2]

$$Z = \frac{R_e(1 + R_i(R_e + R_i)\omega^2 C^2)}{1 + (R_e + R_i)^2\omega^2 C^2} - j\frac{R_e^2 \omega C}{1 + (R_e + R_i)^2\omega^2 C^2}, \tag{1}$$

BIA:ssa käytetään elektrodipareja. Elektrodeilla syötetään herätteena toimiva virta ja mitataan vasteena syntynyt jännite. Kahden elektrodin (bipolaarisessa) konfiguraatiossa haittapuolena on, että samalla parilla joudutaan syöttämään virtaa ja mittaamaan jännitettä samanaikaisesti. Tällöin jännitteen mittausta häiritsee elektrodien polarisaatioimpedanssi, joka aiheutuu muun muassa elektrodin ja elektrolyytin välisestä ioninvaihdosta. Polarisaatioimpedanssin vaikutus on merkittävä virtaa syöttävässä elektrodiparissa, koska se riippuu läpi kulkevan virran suuruudesta. Tämän välttämiseksi käytetään neljän elektrodin (tetrapolaarista) konfiguraatiota, jossa yhdellä parilla syötetään virtaa ja toisella mitataan jännitettä. Jännitettä mittaava elektrodipari sijoitetaan niin, että sen läpi ei kulje merkittävää virtaa, jolloin polarisaatioimpedanssi ei haittaa mittausta. [18]

3 Terttu-terveystuolin arviointi ja uuden järjestelmän ideointi

Aalto-yliopiston Health Factoryn kehittämällä Terttu-terveystuolilla voidaan mitata viittä suuretta, jotka ovat sydänsähkökäyrä, epäinvasiivinen verenpaine, bioimpedanssi, paino ja veren happisaturaatio. EKG ja bioimpedanssi mitataan kahdella elektrodiparilla, jotka ovat kiinnitetty tuolin käsinojiin. Bioimpedanssimittausta käytetään kehon koostumuksen selvittämiseen. Siinä käytetään Blomqvistin et al. (2012, [19]) kehittämää OpenEBI-laitteistoa. Veren happisaturaatio mitataan sormenpäähän kiinnitettävällä pulssioksimetrillä. Pulssioksimetrillä voidaan myös selvittää pulssi sormen kohdalla. Verenpaine mitataan epäinvasiivisesti mansetilla. Paino mitataan neljällä kuormakennolla, jotka ovat kiinni tuolin jaloissa. [2]

EKG-mittauksessa käytetään Corscience EMB1 -OEM-moduulia. Mittauksessa käytettiin neljää elektrodia, joilla muodostettiin I-kytkentä. Elektrodeina käytettiin hopeasta valmistettuja levyjä, jotka kiinnitettiin Terttu-terveystuolin molempiin käsinojiin. EKG-signaalia käytettiin sydänsähkökäyrän tuottamisen lisäksi systolisen verenpaineen estimointiin pulse transit time -menetelmällä (PTT), joka perustuu pulssin siirtymäajan mittaamiseen EKG-antureilla ja fotopletysmografilla. PTT-menetelmää ehdotettiin käytettäväksi auskultatorisen tai oskillometrisen menetelmän sijaan, koska sillä voidaan välttää verenpainemansetin käyttäminen ja mansetin kiinnittämiseen kuluva aika ja vaiva. Pulssioksimetrin ohjainkorttina käytettiin Corsciencen ChipOx-ohjainkorttia. Se valittiin matalan virrankulutuksen perusteella, koska Terttu-terveystuoli on akkukäyttöinen. Anturina käytettiin sormenpäähän kiinnitettävää pulssioksimetriä. Terttu-terveystuolin painon mittaus oli toteutettu nelipistemittauksena. Antureina käytettiin kuorma-antureita (QY Electronics, QL-11A), joiden mittausyläraja on 50 kg. Anturit sijoitettiin tuolin jalkojen alle. Systemaattisen virheen vaikutuksen vähentämiseksi mittausjärjestelmän kalibrointi automatisoitiin. [2]

Kaikki edellä mainitut mittauslaitteet oli kytketty tuoliin kiinnitettyyn tietokoneeseen, jonka LabView-ohjelma käsittelee mittausdataa ja näyttää mitatut arvot. Mittaukset voitiin myös tallentaa tietokoneelle salattuina, jolloin ne olivat käytettävissä vain asianmukaisten käyttäjätunnuksien kautta. Tietokonetta käytettiin kosketusnäytön kautta, koska näppäimistö tai hiiri olisivat olleet epäkäytännöllisiä. Tietokone oli kiinnitetty tuolin selkänojaan, joten sitä voitiin käyttää vain tuolin takaa. Vaikka mittauksen kohde ei voinut toimia samanaikaisesti Terttu-terveystuolin käyttäjänä, sitä ei pidetty merkittävänä haittana, koska mittaustapahtumaan osallistuu joka tapauksessa potilaan lisäksi hoitaja. [2]

Terttu-terveystuoli on suunnattu työkaluksi terveydenhuollon ammattilaisille, joten se arvioinnissa koettiin mielekkääksi hyödyntää terveydenhuollon ammattilaisten mielipidettä. Terveystuolin hyödyllisyyden selvittämiseksi sitä koekäytettiin Vantaan kaupungin geriatrisessa akuuttiyksikössä AKOS1:ssa, jossa hoidetaan ja kuntoutetaan ikääntyneitä potilaita. Koekäytön havainnoijina toimi Laurea-ammattikorkeakoulun opiskelijoita, jotka keräsivät palautetta suoraan osaston hoitajilta. [20] Mekaanisten ja mittausteknisten yksityiskohtien parantamisen lisäksi esitettiin kehitysehdotus

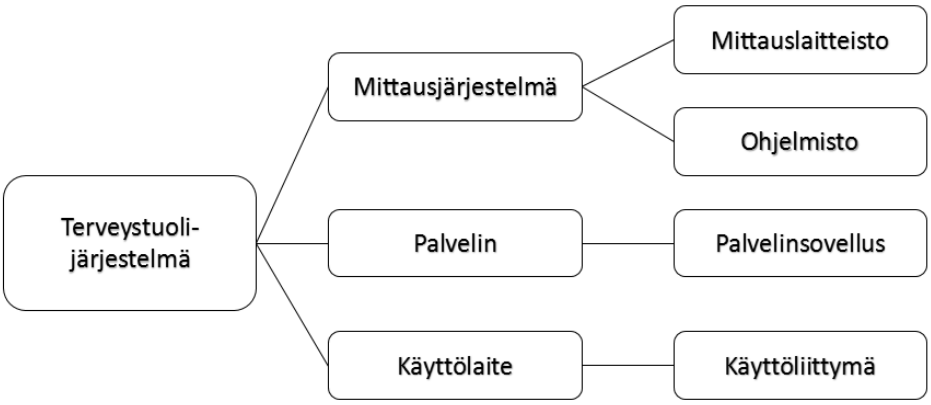
Terttu-terveystuolin liittämisestä potilastietojärjestelmään. Tämän kehitysehdotuksen pohjalta keksittiin joukko muita ideoita, joita käytettiin Terttu-terveystuolin innovoinnissa.

Terttu-terveystuoliin perustuvan innovaation lähtökohdaksi ideoitiin, että osa järjestelmän toiminnoista hajautetaan laitteisiin, jotka sijaitsevat erillään varsinaisesta tuolista. Tämän tarkoittaisi sitä, että tuolissa sijaitseva laitteisto yksinkertaistetaan pelkäksi mittausjärjestelmäksi. Koska mittausjärjestelmän vastuulle jäisi vähemmän tehtäviä, siitä voitaisiin tehdä huomattavasti suppeampi ja yksinkertaisempi. Esimerkiksi mittausten tallennukseen ja esittämiseen liittyvät toiminnot voidaan poistaa siitä kokonaan. Tämä ilmentyisi merkittävänä vähennyksenä järjestelmän valmistuskustannuksissa. Mittaustulokset voidaan lähettää tietoliikenneyhteyden kautta ulkoiselle tietokantapalvelimelle, joka on vastuussa mittausjärjestelmän puuttuvista toiminnoista. Tietokantapalvelin voisi käsitellä mittaustuloksia useasta mittausjärjestelmästä samanaikaisesti, jolloin käyttäjäkapasiteetin skaalaus olisi edullisempaa. Samalla voidaan kerätä useasta eri lähteestä peräisin olevat potilas- ja mittaustiedot yhteen tietokantaan, josta ne ovat helpommin löydettävissä. Viimeinen idea on, että järjestelmää voitaisiin käyttää mobiililaitteeseen asennettavan sovelluksen kautta. Mobiilisovellus antaisi käyttäjälle liikkumavapautta kiinteään tietokoneeseen verrattuna, mukaan lukien mahdollisuuden käyttää mittausjärjestelmää siinä istuessaan. Tämä idean toteutus voisi myös vähentää mittausjärjestelmän valmistuskustannuksia, koska se ei sisältäisi omaa näyttö- tai syötelaitetta.

4 Terveystuoli-järjestelmän suunnittelu

Seuraavissa luvuissa 4.1–4.3 suunnitellaan Terveystuoli-järjestelmän osajärjestelmät, niiden tehtävät, sekä esitellään niihin liittyvä olennainen laitteisto ja ohjelmisto. Suunnittelua voidaan pitää konseptitasoisena, ja tässä diplomityössä ehdotettua toteutusta voidaan muuttaa suunnitteluprosessin jatkuessa, jos ongelmien ratkaisemiseksi havaitaan parempi toteutustapa.

Selkeyden vuoksi Terveystuoli-järjestelmällä tarkoitetaan tästä eteenpäin koko järjestelmää, johon kuuluu kaikki sen osajärjestelmät. Terveystuoli-järjestelmän osajärjestelmiksi määritellään mittausjärjestelmä, palvelinsovellus ja käyttölaite, jotka ovat itsenäisiä ja keskenään kommunikoivia osia. Käyttölaite määritellään tässä diplomityössä laitteeksi, jolla Terveystuoli-järjestelmän käyttöliittymää käytetään. Tässä luvussa suunnitellaan mittausjärjestelmään kuuluva mittauslaitteisto ja ohjelmisto, palvelimella suoritettava palvelinsovellus ja käyttölaitteen käyttöliittymä. Palvelimen fyysistä laitteistoa ei rajata erikseen, vaan palvelinsovellus suunnitellaan toimivaksi tyypillisimmissa palvelinympäristöissä. Käyttölaitteen laitteistoa ei myöskään rajata, koska käyttöliittymän yhteensopivuus halutaan laajaksi. Käsitteistön ja niiden väliset suhteet ovat selvennettynä kuvassa 6.



Kuva 6: Käsitekartta Terveystuoli-järjestelmän osajärjestelmistä

4.1 Mittausjärjestelmä

Mittausjärjestelmä on Terveystuoli-järjestelmän osajärjestelmä, joka on vastuussa mittausdatan keräämisestä, käsittelystä ja lähettämisestä palvelimelle. Se on sulautettu järjestelmä, jonka sisäinen toiminta on käyttäjältä piilossa. Mittausjärjestelmällä ei ole suoraa käyttäjärajapintaa, vaan käyttäjä hallitsee sitä epäsuorasti käyttölaitteen kautta. Käyttäjälle tämä ero näkyy ainoastaan siten, että käyttölaite ei ole fyysisesti kiinni mittausjärjestelmässä. Jatkuvat mittaukset kuten EKG tai pulssioksimetria voidaan aloittaa milloin tahansa käyttölaitteen kautta, ja mittausta voidaan jatkaa niin kauan kun on tarpeen. Verenpainemittauksessa käyttäjän on annettava erillinen komento, koska mansetti on puettava päälle ennen mittauksen aloittamista. Tämä

komento voidaan ohjata Terveystuoli-järjestelmän muiden osajärjestelmien kautta. Komentojen lähettäminen ja mittausdatan vastaanotto edellyttää kaksisuuntaista tiedonsiirtoa. Tässä luvussa valitaan laitteet, joilla mittausdata voidaan kerätä, käsitellä ja lähettää eteenpäin. Laitteiston vaatiman ohjelmiston rakenne suunnitellaan tämän luvun loppuosassa, kun laitteiston mahdollistamat toiminnot ovat esitelty.

Mittauslaitteina on käytettävä laitteita, jotka ovat ohjattavissa sähköisesti. Käyttötarkoituksen kannalta haluttaviin mittauksiin kuuluvat muun muassa EKG, pulssioksimetria, verenpaine, paino ja bioimpedanssi. Kuten luvussa 3 mainittiin, Terttuterveystuolissa aikaisemmissa versioissa käytettiin OEM-moduuleita, jotka mahdollistivat muun muassa mittaussignaalin vahvistuksen ja suodatuksen. OEM-moduulien käytössä on etuna, että ne ovat yleisesti ottaen luotettavia ja hyvin testattuja. Niitä käyttämällä yksinkertaistaa mittausjärjestelmän rakentamista, koska itse toteutettavia komponentteja on vähemmän. OEM-moduuleista saatava data on kuitenkin tulkittava ja muutettava muotoon, jossa sen tallentaminen ja esittäminen on sopivinta. Tähän tarvitaan tietokone ja tietokoneohjelma, joka lukee OEM-moduulin lähettämää dataa sen kommunikaatioprotokollan mukaisesti. Jos sopivaa kommunikaatioprotokollaa käyttävää ohjelmaa ei ole valmiiksi saatavilla, se on kirjoitettava itse. Luvussa 3 asetettujen tavoitteiden mukaan mittausjärjestelmän on oltava myös yhteydessä tietokoneverkkoon, joka muodostuu mittausjärjestelmän lisäksi palvelimesta ja käyttölaitteesta. Tämän takia mittausjärjestelmän on sisällettävä myös verkkotoiminnallisuus.

4.1.1 Mittauslaitteisto

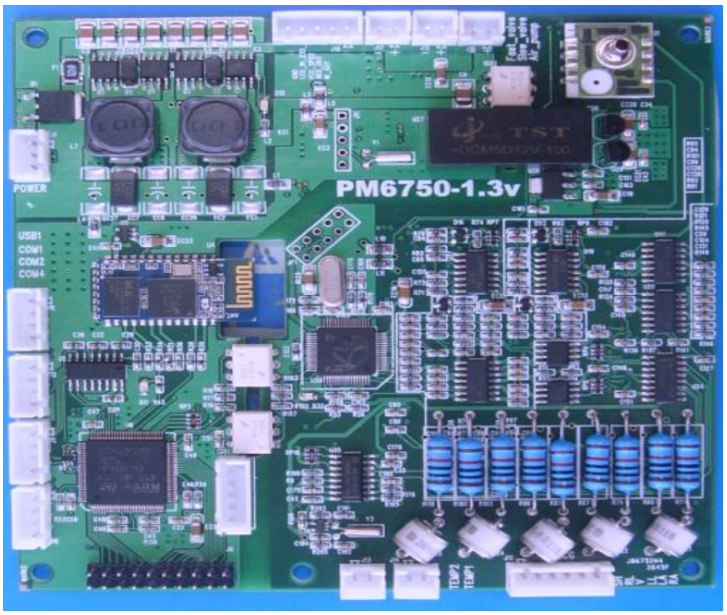
Mittauslaitteistona käytetään OEM-moduuleita, jotka voidaan sellaisenaan liittää osaksi suurempaa järjestelmää. Sovelluksen kannalta halutut mittaukset ovat verenpaine, EKG, pulssioksimetria, paino ja bioimpedanssi. Aalto-yliopiston Health Factory tarjosi käytettäväksi kahta OEM-moduulia, jotka ovat Shanghai Berry Electronic Tech Co. Ltd:n valmistama PM6750 sekä Texas Instrumentsin valmistama AFE4300EVM-PDK. Ensiksi mainittu mahdollistaa epäinvasiivisen verenpainemittauksen, EKG:n ja pulssioksimetrian. Jälkimmäiseksi mainittu on suunniteltu painon ja bioimpedanssin mittaukseen.

PM6750

PM6750 Bluetooth Six-parameter Patient Monitor Module on useaa parametria mittaava OEM-moduuli, joka on tarkoitettu potilasseurantaan. Se on CE-merkitty ja sitä markkinoidaan Euroopassa, Amerikassa ja Aasiassa. PM6750:n on suunnittelija ja valmistaja on lääketieteen tekniikan yritys Shanghai Berry Electronic Tech Co., Ltd. PM6750 voidaan liittää tietokoneeseen USB- tai RS-232-liitännällä. Laitteeseen on saatavilla Windows-yhteensopivia laiteajureita. Se myös sisältää Bluetooth-moduulin, jolla voidaan muodostaa langaton yhteys fyysisen liitännän sijaan. [21]

PM6750:n soveltuvuutta terveystuolin mittausjärjestelmään arvioitiin tutkimalla sen liitettävyyttä. PM6750:n kotelossa on läpiviennit B-tyypin USB-liittimelle ja RS-232 DE-9 liittimelle. Kotelon sisäpuolella olevan piirilevyn vasemmalla reunalla on myös pinnejä, joilta lähtösignaalit voidaan myös kerätä (ks. kuva 7). Piirilevyn pinnien

käyttö voi olla tarpeellista, jos USB- tai RS-232-kaapeleita ei voida käyttää. PM6750:n liitännässä voidaan tarvittaessa käyttää myös TTL-liitäntää. Piirilevyllä on MAX232-piiri, jolla muutetaan RS-232-portin signaaleja TTL-yhteensopivaan muotoon. RS-232- ja TTL-sarjayhteyksien liittämisessä ulkoiseen laitteeseen on huomioitava, että ne käyttävät eri jännitetasoja. RS-232-standardin mukaan sen korkein jännitetaso on 15 V maatasoon nähden [22], mikä voi olla joillekin mikrokontrollereille liian suuri. Silloin TTL:n käyttäminen voi olla välttämätöntä, koska sen jännitetaso on vain 5 V. Sekä RS-232 ja TTL käyttävät samaa fyysistä liitintä PM6750:n kuoren ulkopuolella. TTL:n tai RS-232:n käyttö voidaan valita vaihtamalla jumpperin paikkaa PM6750:n piirilevyllä.



Kuva 7: PM6750:n piirilevy ilman liitettyjä antureita. [21]

Kokonaisuudessaan PM6750:n liitettävyyttä voidaan pitää hyvänä, koska USB-liitännät ovat tietokoneissa yleisiä. Laitteen valmistaja tarjoaa laiteajureita vain Windows-järjestelmille, mutta tämä rajoitus voidaan ohittaa käyttämällä RS-232-porttia ja sopivaa USB-sarja-adapteria. Adapterin kanssa voidaan käyttää adapterin valmistajan tarjoamia laiteajureita, jotka ovat usein Windows-järjestelmien lisäksi yhteensopivia myös Linux-pohjaisissa järjestelmissä.

PM6750-moduuliin voidaan liittää useita antureita. Yhteensopiviin antureihin kuuluu verenpainemansetti, pulssioksimetri, viisi EKG-elektrodia sekä lämpömittari. Kaikki yhteensopivat anturit olivat sisällytetty samaan pakkaukseen kuin PM6750, joten niitä ei ollut tarpeellista hankkia erikseen. Mansetti on oskillometrinen ja sopivan kokoinen aikuisen puettavaksi. Sitä voidaan käyttää suoraan Terveystuoli-järjestelmässä. Mansetin ilmapussi täytetään ainoastaan PM6750:n kautta, mitä varten sille on lähetettävä komento sarjaväylän kautta. Pulssioksimetri on sormeen kiinnitettävä. Siinä ei ole painikkeita tai näyttöä, ja sitä luetaan ainoastaan PM6750:n kautta. Lämpömittari on kehon sisäiseen ja ulkoiseen käyttöön tarkoitettu termistori. [21]

EKG-elektrodeissa on adhesiivinen Ag/AgCl-geelipinta kiinnitystä varten. Adhesiivista kiinnitystä halutaan kuitenkin välttää, koska kiinnitys viivästyttää mittauksen aloittamista, ja elektrodin irrottaminen voi aiheuttaa epämukavuutta potilaalle. Yleisesti käytetyt EKG-elektrodit ovat kertakäyttöisiä, ja niiden vaihtaminen useasti vähentäisi mittausjärjestelmän käytännöllisyyttä. Niiden korvaajiksi on kuitenkin esitetty vaihtoehtoja. Esimerkkinä voidaan mainita Lim et al. (2006, [23]), jotka käyttivät tuoliin kiinnitettyjä aktiivielektrodeja EKG:n mittaamiseen. Terttu-terveystuolissa taas käytettiin tuolin käsinojiin kiinnitettyjä hopeasta valmistettuja kuivaelektrodeja, joiden tuottaman signaalin laatu oli tyydyttävä [2]. Tällä perusteella kuivaelektrodien käyttö voidaan sisällyttää Terveystuoli-järjestelmään, ainakin siihen asti kunnes parempi ratkaisu löydetään.

PM6750:n tiedonsiirto on sarjamuotoista. Sarjaväylä lähettää ja vastaanottaa paketteja, jotka koostuvat neljästä kehystavusta ja data-osiosta, jonka pituus voi vaihdella. Paketti sisältää dataa vain yhdestä mittaustyypistä kerrallaan, ja eri mittaukset tuottavat eri määrän paketteja sekunnissa. EKG-aaltoa sisältäviä paketteja lähetetään 250 kpl/s, pletysmografia-aaltoa 40 kpl/s ja muita mittausparametreja 1–2 kpl/s. Paketteja lähetetään yhteensä enintään 345 kpl/s, mutta usein harvemmin. Verenpainedataa lähetetään ainoastaan verenpainemittauksen aikana. [21]

Paketti alkaa kahden tavun muodostamalla aloitusmerkillä, jonka jälkeen yksi tavu kertoo paketin koon. Aloitusmerkin ja paketin koon jälkeen lähetetään paketin dataosio. Kullakin mittauksella on erilainen data-osio, joten ne tulee käsitellä eri tavoin. Saman paketin dataosio voi sisältää useita. Eri tyyppisien pakettien dataosioita selventää kuva 8. Paketin viimeinen tavu on tarkistussumma, jonka pituus on yksi tavu. PM6750:n dokumentaatio kuvaa kaikki tavut heksadesimaalimuodossa. Kommunikaatioprotokolla on kuvattu yksikäsitteisesti, ja pakettien sisältö voidaan tulkita tietokoneohjelmalla. Laitteen lähettämät ja vastaanottamat paketit käyttävät samaa protokollaa. [21]

TYPE	A1	A2	A3	A4	A5	A6	A7	A8	Freq (Pkg/sec)
ECG Wave	0x01	I	II	III	aVR	aVL	aVF	V	250
ECG Param	0x02	ECG Status	HeartRate	RespRate	ST Level	ARR code			1
NIBP Param	0x03	NIBP Status	Cuff Pressure	Sys Pressure	Mean Pressure	Dia Pressure			2
SPO2 Param	0x04	SPO2 Status	Spo2Sat	PulseRate					1
TEMP Param	0x05	TEMP Status	TEMP1 Integral	TEMP1 Decimal	TEMP2 Integral	TEMP2 Decimal			1
Software Version	0xFC	ASCII byte 1	ASCII byte 2	ASCII byte 3	···	ASCII byte n	···	···	N/A
Hardware Version	0xFD	ASCII byte 1	ASCII byte 2	ASCII byte 3	···	ASCII byte n	···	···	N/A
SPO2 Wave	0xFE	Wave amplitude							40
RESP Wave	0xFF	Wave amplitude							50

Kuva 8: PM6750:n lähettämien pakettien dataosuus. [21]

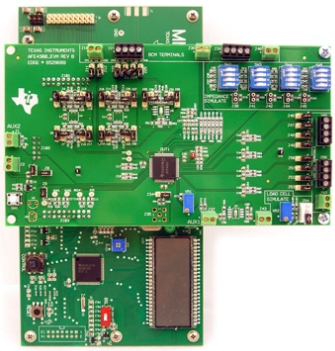
PM6750:n lähettämästä datasta ei ole tarpeellista kerätä kaikkia parametreja. Datapaketteja on yhdeksää eri tyyppiä, mutta vain viisi niistä ovat kiinnostavia. Kuvan 8 mukaisesti halutut tyypit ovat ECG Wave, ECG Param, NIBP Param, SPO2 Param ja SPO2 Wave, jotka sisältävät tietoa EKG:sta, verenpaineesta ja pulssioksimetriasta. Esimerkiksi lämpötila (Temp) tai laitteen version numerot eivät ole tarpeellisia. Näiden datapakettien sisällöstä ei myöskaan ole tarpeellista kerätä

kaikkea sisältöä. Esimerkiksi EKG-pakettien parametreistä kerätään vain A2 (I-kytkentä), jos muista kytkennöistä ei olla kiinnostuneita.

AFE4300EVM-PDK

AFE4300EVM-PDK on sovelluskohtainen mikropiiri, joka on kehitetty painon ja bioimpedanssin mittaukseen. Se sisältää signaaliketjut molemmille mittauksille. Signaaliketjut käyttävät yhteistä 16-bittistä delta-sigma-muunninta, jonka baudinopeus on enimmillään 860 Bd. Koska AD-muunnin on molemmille mittauksille yhteinen, mittauksia ei voida suorittaa samanaikaisesti. Tämän rajoituksen ei kuitenkaan pitäisi vaikuttaa mittausjärjestelmän toimintaan käytännössä, koska mittauksia voidaan suorittaa vuorotellen.

AFE4300EVM-PDK on evaluaatioversio Texas Instrumentsin AFE4300:sta, joka on tarkoitettu samaan käyttöön. EVM-PDK on koostuu kahdesta piirilevystä, jotka ovat pinottu päällekkäin (ks. kuva 9). [27] Ylempi piirilevy sisältää anturiliitännät, SPI-liitännän, AD-muuntimen ja muun signaalinkäsittelyyn tarvittavan laitteiston. Alempi piirilevy vaikuttaa olevan Texas Instrumentsin digitaalinen ohjainpiiri, joka tuottaa SPI-väylän ja signaaliprosessorin käyttämän kellosignaalin. Se sisältää myös USB-liitännän, jota ylemmässä piirilevyssä ei ole. Koon pienentämiseksi on mahdollista poistaa laitteen alempi piirilevy ja käyttää ylempää piirilevyä terveystuolin mittausjärjestelmässä. Tällöin SPI-väylälle ja signaaliprosessorille on syötettävä kellosignaali jollain muulla laitteella.



Kuva 9: AFE4300EVM-PDK:n piirilevyt ilman antureita. [24]

AFE4300EVM-PDK kommunikoi ulkoisten laitteiden kanssa SPI-väylän kautta. Sen avulla voidaan lukea signaaliprosessorin muuntamaa dataa, kirjoittaa ja lukea laitteen rekisteriä sekä antaa laitteelle komentoja. Sen rajapinta käyttää neljää signaalia, jotka ovat kellosignaali, datalinja ulkoisesta laitteesta signaaliprosessoriin, datalinja signaaliprosessorista ulkoiseen laitteeseen ja ulkoisen laitteen valintasignaali. Sarjaväylän signaalien lisäksi on käytössä laitteen sisäinen signaali, joka ilmoittaa milloin anturidatan muunnos on valmis. Data siirtyy 24 bitin paketteina, josta kahdeksan ensimmäistä bittiä ovat kehysbittejä ja loput 16 ovat databittejä. Paketin merkitsevin bitti siirtyy ensin. Kehysbiteistä viimeiset viisi ovat osoitebittejä, jotka

viittaavat haluttuun osoitteeseen signaaliprosessorin rekisterissä. Bitti 21 määrää luetaanko vai kirjoitetaanko dataa. [27]

AFE4300EVM-PDK:n AD-muunnin voi toimia jatkuvassa tilassa tai kertamuunnostilassa. Jatkuvassa tilassa uusi mittaussignaali muunnetaan ja siirretään muunnosrekisteriin heti kun edellinen muunnos on saatu valmiiksi. Kertamuunnostilassa odotetaan kunnes AD-muunninta vastaavaa rekisteriä muutetaan. Kertamuunnostilaa käyttämällä voidaan vähentää tehonkulutusta jatkuvaan tilaan verrattuna. Tiloja voidaan vaihtaa lähettämällä sille komentoja SPI-yhteyden kautta, esimerkiksi mikrokontrollerin avulla. [27]

4.1.2 Tietokoneen valinta

Terttu-terveystuolissa käytettiin kosketusnäytöllistä henkilökohtaista tietokonetta. Tietokoneessa oli Windows-käyttöjärjestelmä, ja mittaustulokset kerättiin, käsiteltiin ja esitettiin LabView-ohjelmistolla. Vaikka tämä ratkaisu oli toimiva, sillä muutamia haittapuolia. Windows-käyttöjärjestelmän ja LabView-ohjelmiston käynnistämisessä kului yleensä useita minuutteja aikaa. Ajoittain käytön aikana käyttöjärjestelmän havaittiin asentavan päivityksiä ja käynnistyvän uudelleen, mikä keskeytti mittauksen. Nämä olivat vähäisiä ongelmia, eikä niistä ollut merkittävää haittaa Terttu-terveystuolin käyttämisessä. Huomattavampana seikkana voidaan kuitenkin pitää sitä lähtökohtaa, että Terttu-terveystuolin laitteisto ja ohjelmisto eivät ole sen tarpeisiin nähden kustannustehokkaita. Suuri osa Terttu-terveystuolin hinnasta muodostuu sen käyttämästä henkilökohtaisesta tietokoneesta ja kaupallisesta ohjelmistosta, jotka ovat korvattavissa jollain muulla ratkaisulla. Järjestelmän kehittämisen ja mahdollisen tuotteistamisen kannalta olisi myös toivottavaa vähentää omisteisen ohjelmiston käyttöä. Näiden seikkojen takia on perusteltua harkita toisenlaisen tietokoneen käyttämistä Terveystuoli-järjestelmässä.

Mittausjärjestelmän tietokone tulee suorittamaan hyvin pientä määrää sovelluksia, joten yksinkertaisen ja edullisen tietokoneen, esimerkiksi mikrokontrollerin, käyttö on mahdollista. Mikrokontrollerit ovat pienikokoisia ja edullisia tietokoneita, jotka ovat yleisiä sulautetuissa järjestelmissä. Ne soveltuvat hyvin mittausdatan keräämiseen. Mikrokontrollerin käyttämiseksi voidaan suunnitella oma piirilevy, tai käyttää valmista kehitysalustaa, jossa on jo valmiiksi integroitu kaikki mittausjärjestelmän kannalta oleellinen laitteisto. Terveystuolin mittausjärjestelmään sopivan kehitysalustan valinnassa on huomoitava useita ominaisuuksia. Kehitysalustan mikrokontrollerin suorituskyvyn on oltava riittävä siihen, että OEM-moduuleilta vastaanotettua dataa voidaan käsitellä nopeammin kuin sitä vastaanotetaan. PM6750:een liittämistä varten sillä on oltava RS-232-, Bluetooth- tai USB-liitettävyys. SPI-yhteys on myös edellytys, että se on liitettävissä AFE4300EVM-PDK:n kanssa. Viimeisenä vaatimuksena sen on kyettävä langattomaan yhteyteen palvelinlaitteen kanssa.

Sopivan kehitysalustan etsiminen aloitettiin Arduino-kehitysalustojen eri mallien joukosta, koska ne olivat havaittu hyödyllisiksi Aalto-yliopiston Health Factoryn aikaisemmissa projekteissa, mukaan lukien Terttu-terveystuolissa. Arduino on avoimen lähdekoodin elektroniikka-alusta, jonka laitteisto ja ohjelmointi on tarkoitettu helpokäyttöiseksi. Arduinosta on saatavilla useita eri malleja, ja niitä on käytetty hyvin

laajassa määrässä erilaisia sovelluksia, mukaan lukien valaistustekniikassa, robotiikassa ja kotiautomaatiossa. Useimmat Arduino-kehitysalustat sisältävät AVR-pohjaisen mikroprosessorin, flash- SRAM- ja EEPROM-muistit ja useita analogisia ja digitaalisia pinnejä input/output-liitäntöjä varten. Eräät mallit sisältävät myös WLAN- tai Bluetooth-moduulin, Linux-pohjaisen käyttöjärjestelmän, ARM-prosessorin tai sisäänrakennettuja antureita. Arduino-laitteet ovat itsenäisesti toimiva kehitysalustoja, joiden käyttöön ei tarvita muita komponentteja kuten muisteja tai kellopiirejä. Arduinoon on saatavilla suuri määrä erilaisia laajennuslaitteita (engl. shield), jotka ovat suunniteltu toimintojen ja liitettävyyden lisäämiseksi. Niitä voidaan pinota päällekäin, mikä mahdollistaa usean laajennuslaitteen käytön samanaikaisesti. Laajennuslaitteiden avulla Arduinoon voidaan liittää esimerkiksi näppäimistöjä, digitaalikameroita tai massamuistilaitteita. Nämä ominaisuudet huomioiden Arduino vaikuttaa riittävän hyvältä tässä diplomityössä käytettäväksi, koska sillä voidaan toteuttaa anturidatan kerääminen ja kommunikaatio järjestelmän muiden osien kanssa. Arduino-kehitysalustojen käyttöön vaikutti oleellisesti myös Aalto-yliopiston Health Factoryn aikaisempi kokemus niistä. Arduinon sijaan mittausjärjestelmän tietokoneeksi olisi mahdollisesti kelvannut jokin muukin kehitysalusta, kuten Raspberry Pi (Raspberry Pi Foundation) tai BeagleBone Black (BeagleBoard.org Foundation).

Arduino Yun

Arduino Yun on kehitysalusta, jonka erityisominaisuuksiin kuuluvat Linux-ympäristöä suorittava prosessori ja sisäänrakennettu Wi-Fi-toiminnallisuus. Useiden muiden Arduino-laitteiden tavoin Arduino Yun sisältää AVR ATmega 32U4 -mikrokontrollerin. Arduino Yunin input/output-pinnit ovat yhdistetty mikrokontrolleriin, jota voidaan käyttää pinnien lukemiseen ja ohjaamiseen. 32U4 käyttää muokattua versiota avoimen lähdekoodin Wiring-ohjelmistokehyksestä. Arduinon Wiring-ympäristöä ohjelmoidaan omalla ohjelmointikielellä, joka on muunneltu versio C- ja C++-kielistä. Ohjelmoinnissa voidaan käyttää Arduino IDE -ohjelmistoa, joka on avoimen lähdekoodin kehitysympäristö. Arduino IDE kääntää lähdekoodin ohjelmatiedostoksi ja lähettää sen mikrokontrollerin käynnistyslataajalle. Käynnistyslataajaa ja sille lähetettyä ohjelmaa säilytetään Arduinon Flash-muistissa. Ohjelma suoritetaan aina automaattisesti Arduinon käynnistyessä. Arduinon ohjelmoinnissa voidaan käyttää myös muita kehitysympäristöjä jotka ovat yhteensopivia C:n ja C++:n kanssa. Ohjelmien lataamisessa Arduino Yunille käytetään siinä olevaa Micro-B USB-porttia. Portti toimii USB-laitteena, joten sitä ei voida käyttää lisälaitteiden liittämiseen, jotka tarvitsevat USB-isäntä-porttia. Toiseen tietokoneeseen liitettynä Micro-B USB-porttia voidaan käyttää myös input/output-väylänä tietokoneen ja mikrokontrollerin välillä. Arduino ottaa tarvitsemansa käyttöjännitteen (5 V) tämän USB-portin kautta. [25]

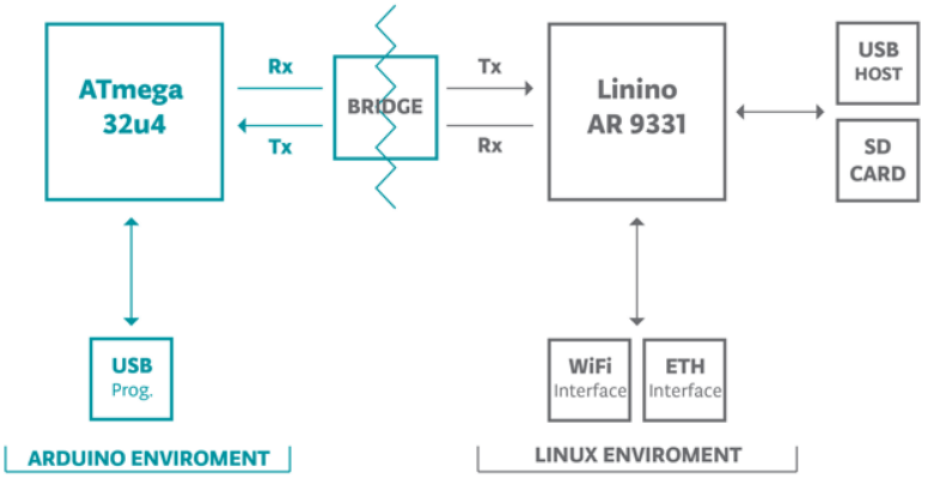
Arduino Yun sisältää ATmega 32U4 -mikrokontrollerin lisäksi Atheros AR9331 -prosessorin, joka on yhdistetty Ethernet- ja USB-A-portteihin ja Wi-Fi-moduuliin. AR9331 ei ole suoraan yhdistetty Arduinon I/O-pinneihin kuten 32U4, mutta se voi lukea ja ohjata niitä AR9331:n ja 32U4:n välisen sisäisen sarjaväylän kautta. Sarjaväylän käyttöön on olemassa valmiita funktioita sisältävä ohjelmakirjasto, mutta myös oman rajapinnan toteuttaminen on mahdollista. AR9331:n suorittamalla

järjestelmällä on oma fyysinen USB-A-portti, mutta toisin kuin 32U4:n mikrokontrolleriympäristö, se toimii USB-isäntänä. Arduino Yunin sisäänrakennettu Wi-Fi mahdollistaa langattoman verkkoyhteyden muodostamisen. Wi-Fiä voidaan käyttää tukiasemana tai se voidaan yhdistää olemassa olevaan verkkoon. Verkkoyhteys ei ole suoraan yhdistetty 32U4-mikrokontrolleriin, mutta sitä voidaan käyttää 32U4:n ja AR9331:n välisen sarjaväylän kautta. Verkkoyhteys voidaan konfiguroida komentorivin kautta, mutta myös verkkopohjaisen rajapinnan kautta (LuCi), jota voidaan käyttää internetselaimella muodostamalla HTTP-yhteys Arduino Yunin IP-osoitteeseen. [25]

Atheros AR9331 tukee OpenWrt-pohjaista Linux-levitysversiota OpenWrt-Yunia. OpenWrt-Yunia voidaan käytetään Unix-komentorivin kautta. Komentoriviä voidaan käyttää mikrokontrollerin kautta hyödyntämällä laitteen sisäistä sarjaväylää, tai muodostamalla etäyhteys Arduino Yuniin SSH-protokollalla. OpenWrt-Yun-käyttöjärjestelmä käyttää 9 MB Arduino Yunin sisäisestä 16 MB:n flash-muistista. Haihtumatonta muistia voidaan laajentaa käyttämällä Arduino Yunin microSD-korttipaikkaa. Linux-ympäristöön on valmiiksi asennettu muun muassa Python 2.7 ja opkg. Opkg on paketinhallintaohjelma, joka on kehitetty Linux-pohjaisille sulautetuille järjestelmille. Opkg:lla voidaan asentaa muun muassa sovelluksia ja ajureita, jotka ovat rakennettu opkg-paketeiksi. Opkg-paketteja voidaan hakea tiedostovarastoista (engl. repository), joita ylläpidetään internetpalvelimilla. OpenWrt-Yunille on koottu oma tiedostovarasto osoitteessa https://github.com/arduino/openwrt-packages-yun (tarkistettu 1.9.2015). [25]

Arduino Yunin Linux-ympäristön ohjelmoiminen eroaa merkittävästi 32U4:n ohjelmoimisesta. AR9331 on mikroprosessori, joten kaikki sillä suoritettavat ohjelmat käynnistetään käyttöjärjestelmän kautta. OpenWrt-Yunin Python 2.7 -tuki mahdollistaa Python-skriptien suorittamisen. Python-skripti voidaan kirjoittaa ja tallentaa suoraan OpenWrt-Yunissa tai siirtää sinne toiselta laitteelta. Skripti voidaan käynnistää muiden Linux-levitysversioiden tapaan komentorivin kautta. Python-skripteillä voidaan käyttää Linux-ympäristön resursseja, kuten Wi-Fi-yhteyttä, SD-korttia tai USB-isäntä-porttia. Vastaavasti kyseiset resurssit eivät ole suoraan Arduino Yunin 32U4-mikrokontrollerin käytettävissä. [25] Resurssien jakautumista havainnollistaa kuva 10

Yhteenvetona Arduino Yunin kyvyistä voidaan mainita, että se täyttää mittausjärjestelmän vaatimukset. Yuin mikrokontrolleri voidaan ohjelmoida kommunikoimaan AFE4300EVM-PDK:n kanssa. Yunin Linux-ympäristön ja laitteisto ja ohjelmisto soveltuu USB-portin lukemiseen ja kirjoittamiseen, mitä tarvitaan PM6750:n ohjauksessa. Sisäänrakennettu Wi-Fi mahdollistaa verkkoyhteyden muodostamisen mittausjärjestelmän ja palvelimen kanssa. Nämä ominaisuudet olivat ratkaisevia Arduino Yunin valinnassa Terveystuoli-järjestelmän mittausjärjestelmän tietokoneeksi. Valintaan vaikutti myös kehitysalustan helppokäyttöisyys ja olemassa olevan dokumentaation ja tuen määrä. Arduinon verkkosivu sisältää erilaisia käyttöoppaita, dokumentaation Arduino IDE:n sisäänrakennetuista ohjelmistokirjastoista ja tarjoaa useita esimerkkiohjelmia, joita voi käyttää hyödykseen omissa ohjelmissa.



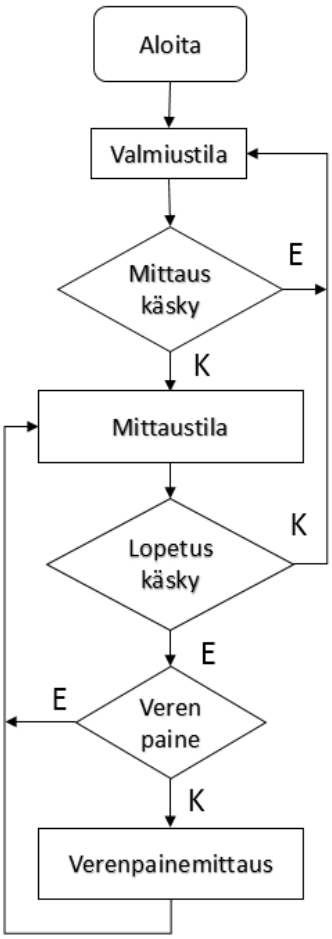
Kuva 10: Arduino Yunin mikrokontrolleriympäristö, Linux-ympäristö ja niitä yhdistävä Bridge-väylä. [25]

4.1.3 Mittausjärjestelmän toiminnallinen määrittely

Mittausjärjestelmä toimii useassa eri tilassa. Tiloiksi määritellään valmiustila, mittaustila ja verenpainemittaustila. Mittausjärjestelmä käynnistyessään muodostaa verkkoyhteyden palvelinsovelluksen kanssa ja siirtyy valmiustilaan. Mittausjärjestelmä pysyy valmiustilassa kunnes se saa käskyn palvelinsovellukselta. Kun käyttäjä haluaa aloittaa mittauksen, palvelin välittää mittauskäskyn järjestelmälle, joka siirtyy mittaustilaan. Mittaustilassa järjestelmä on yhden käyttäjän varaama, eikä sillä voi olla muita käyttäjiä. Jatkuvat mittaukset kuten EKG-mittaus ja pulssioksimetria suoritetaan automaattisesti. Verenpainemittaus aloitetaan, kun järjestelmä on mittaustilassa ja käyttäjä antaa käskyn verenpainemittauksen aloittamisesta. Verenpainemittaus kestää usean sekunnin ajan, jonka jälkeen järjestelmä siirtyy takaisin mittaustilaan. Kun käyttäjä on valmis, hän lähettää käskyn mittauksen lopettamisesta. Palvelin välittää lopetuskäskyn mittausjärjestelmälle, joka siirtyy takaisin valmiustilaan. Tämä toiminta on kuvattu vuokaaviona kuvassa 11.

4.2 Palvelinsovellus

Terveystuoli-järjestelmää suunniteltaessa päätettiin, että osa järjestelmän toiminnoista hajautetaan tietokoneverkkoon, jolloin mittausjärjestelmä voidaan pitää mahdollisimman yksinkertaisena. Palvelin on yksi Terveystuoli-järjestelmän kolmesta osajärjestelmästä. Palvelimella on kolme tehtävää. Ensimmäinen tehtävä on kommunikoida mittausjärjestelmän kanssa vastaanottaen mittausdataa ja lähettäen komentoja mittauksen aloittamisesta. Toinen tehtävä on tallentaa mittausdataa, jotta sitä voidaan tarkastella myöhemmin. Kolmas tehtävä on välittää mittausdataa ja -komentoja käyttölaitteiden ja mittausjärjestelmän välillä. Nämä tehtävät edellyttävät, että palvelin on aina mittausjärjestelmän ja käyttäjän tavoitettavissa verkkoyhteyden yli.



Kuva 11: Mittausjärjestelmän tilat ja niiden vaihtuminen.

Palvelin voidaan suunnitella käytettäväksi laajaverkossa, kuten internetissä, tai rajoittaa sen käyttö pienempään verkkoon, kuten lähiverkkoon. Eri verkkojen käytössä on niille ominaisia etuja ja haittoja, ja on sovelluskohtaista, mitkä edut tai haitat ovat merkittäviä. Laajin käytettävyys voidaan kuitenkin saavuttaa jos järjestelmä suunnitellaan yhteensopivaksi internetyhteyden kanssa. Internetin yli toimiakseen palvelimen on käytettävä Internet-protokollaa.

4.2.1 Internet-protokollan käyttö palvelinsovelluksessa

TCP/IP on internetyhteyden edellyttämä joukko protokollia, jotka määräävät miten kaksi tietokonetta kommunikoivat keskenään. Tiedonsiirrossa verkon yli käytetään paketteja, jotka ovat tarkoitettu tietylle IP-osoitteelle. IP-osoitteen pituus on rajoitettu, joten eri osoitteita on vain rajallinen määrä. IPv6-protokolla käyttää 128-bittistä osoitetta, mikä rajaa mahdollisten IPv4-osoitteiden määrän 2^{128} kappaleeseen. Vaikka määrä on suuri, internetpalveluntarjoajilla on käytössään vain pieni osa koko osoiteavaruudesta. Ellei palvelintarjoajan kanssa ole toisin sovittu, asiakkaan IP-osoite

vaihdetaan säännöllisesti. Palvelimen ylläpitäminen edellyttää, että IP-osoitteiden vaihtuminen huomioidaan. Osoitteen vaihtuminen voidaan välttää varaamalla kiinteä IP-osoite, joka ei vaihdu automaattisesti. Kiinteä IP-osoite voidaan esimerkiksi ostaa internetpalveluntarjoajalta, ja usein kyseistä palvelua tarjotaan vain yritysasiakkaille. Kiinteän IP-osoitteen hankkimiselle on muitakin vaihtoehtoja. Useat yritykset tarjoavat esimerkiksi palvelinpaikkoja yrityksen omistamassa konesalissa, tai palvelinkapasiteetin vuokrausta yrityksen omistamalta palvelimelta. Tällöin konesalin tai palvelimen ylläpitäjä on vastuussa, että yhteysyritykset päätyvät oikealle tietokoneelle. Palvelinsovelluksen pitkäaikainen ylläpito edellyttää jonkin näistä ratkaisuista, joilla pääsy verkkopalveluun voidaan taata. [26]

TCP-yhteys on yksi TCP/IP-protokollan mahdollistamista yhteysmuodoista. Se voidaan muodostaa, jos kohteen IP-osoite ja porttinumero tiedetään. Porttinumero on 16-bittinen luku, jota käytetään IP-osoitteen tavoin kohteen osoitteena. Siinä missä IP-osoitetta voidaan pitää tietokoneen osoitteena verkossa, porttinumeroa voidaan pitää prosessin osoitteena tietokoneessa. IP-osoite ja porttinumero muodostavat yhteyden pääpisteen, josta voidaan käyttää nimeä pistoke (engl. socket). Pistoke sisältää myös tiedon käytetäänkö sitä TCP- vai UDP-protokollan mukaisesti. Pistokkeiden käyttötarkoitukset ohjelmoinnissa, ja niiden mahdollistamat toiminnot, voivat vaihdella eri ohjelmointikielien välillä. Vaikka pistoke on määritelmänsä mukaisesti tunniste, sillä voidaan tarkoittaa myös muita asioita eri asiayhteyksissä. Esimerkiksi olio-ohjelmoinnissa pistokkeiksi voidaan kutsua olioita, joiden kautta voidaan lähettää ja vastaanottaa viestejä. [26]

Yhteyden aloittava sovellus luo pistokkeen, jolle annetaan tiedot kohteen IP-osoitteesta ja porttinumerosta. Jos pistokkeen kautta yritetään ottaa yhteys palvelimelle, palvelin voi kirjata yhteyttä yrittävän tietokoneen IP-osoitteen ja porttinumeron, jonka kautta yhteydenottoyritys saapui. Palvelin voi muodostaa näiden tietojen perusteella uuden pistokkeen, jolloin palvelinsovelluksen ja asiakassovelluksen välillä on kaksisuuntainen yhteys. Yhteydenottoyrityksien odottamista kutsutaan kuuntelemiseksi. Kuunteleminen sidotaan tiettyyn porttinumeroon tai -numeroalueeseen, jotta data voidaan osoittaa vain tarkoituksenmukaisille sovelluksille. Jos kaksi sovellusta käyttää samaa porttinumeroa, ongelmatilanteita voi muodostua. [26]

4.2.2 Ohjelmointikielen ja ohjelmistoalustan valinta

TCP:tä käyttävä palvelinsovellus on mahdollista kehittää usealla eri ohjelmointikielellä ja yhteensopivaksi erilaisissa tietokoneympäristöissä. Koska TCP-yhteyden käyttäminen ei ole rajattu millekään yhdelle ympäristölle, niiden joukosta voidaan valita käytännöllisin. Käytännöllisimmäksi vaihtoehdoksi todettiin Ryan Dahlin kehittämä Node.js-ohjelmistoalusta [31], koska siitä oli jo valmiiksi kokemusta. Node.js on tarkoitettu palvelinsovellusten kehittämiseen varsinkin sellaisiin tilanteisiin, joissa input/output-operaatioihin kuluva aika halutaan minimoida [28]. Koska mittausjärjestelmä tuottaa ja lähettää mittauspisteitä yli 250 Hz:n taajuudella, palvelinsovelluksen on suoritettava keskimäärin yhtä monta input-operaatiota sekunnissa. Jos mittauspisteet halutaan lähettää mahdollisimman pienellä viiveellä käyttölaitteelle, myös output-operaatioiden määrä on yhtä suuri.

JavaScript soveltuu hyvin palvelinsovelluksessa käytettäväksi ohjelmointikieleksi, koska se käyttää ns. asynkronisia palvelupyyntöjä. Tämä tarkoittaa että kun palvelupyynnön tarjoavaa funktiota kutsutaan, sitä ei suoriteta välittömästi, vaan vastauksena jollekin toiselle tapahtumalle. Tapahtuma voi olla esimerkiksi yhteydenotto tai painikkeen klikkaaminen, ja se voi tapahtua ohjelman kannalta milloin tahansa. Asynkronisten palvelupyyntöjen parametrina annetaan callback-funktio, jossa määritellään miten tapahtumaan vastataan. Asynkronisen toiminnan etuna on, että ohjelma voi jatkaa muuta toimintaansa sillä aikaa kun se odottaa palvelupyynnön toteutumista. Palvelupyynnöillä voidaan toteuttaa esimerkiksi TCP-yhteyksien muodostaminen. Esimerkkinä yksinkertaisesta palvelinsovelluksesta voidaan esittää seuraavaa tilannetta:

- palvelinsovellus käynnistyy ja alkaa kuuntelemaan ennalta määrättyä porttia
- tuleva TCP-yhteys havaitaan tapahtumana, ja tapahtuma käynnistää palvelupyynnön
- pyynnön seurauksena ohjelma hyväksyy TCP-yhteyden ja luo sitä vastaavan pistokkeen

Koska tulevia yhteyksiä käsitellään asynkronisesti, ne eivät keskeytä portin kuuntelua muiden yhteyksien varalta.

Python- ja JavaScript-kielissä pistokkeet ovat olioita, jotka lähettävät ja vastaanottavat dataa TCP-yhteyden yli metodikutsuilla. Node.js:aa käyttäen portin kuuntelu voidaan aloittaa listen()-metodilla, joka toimii asynkronisesti. Tämän jälkeen Python-sovelluksella voidaan aloittaa yhteyden connect()-metodilla. Pistokkeiden yhdistämisen jälkeen asiakas ja palvelin toimivat tasavertaisina, ja molemmat puolet voivat lähettää ja vastaanottaa dataa vuorotellen tai samanaikaisesti. Pythonissa datan lähettäminen ja vastaanottaminen voidaan toteuttaa send()- ja recv()-kutsuilla. Node.js:ssa käytetään vastaavasti asynkronisia palvelupyyntöjä.

4.2.3 Palvelinsovelluksen ylläpito

Kuten yllä on mainittu, mittausjärjestelmä ja palvelinsovellus voidaan yhdistää käyttäen TCP-protokollaa. Tällöin on kuitenkin huomioitava, että IP-osoitteet eivät normaalisti ole pysyviä ellei internetpalveluntarjoajalta osteta kiinteää IP-osoitetta. Terveystuoli-järjestelmän tulevissa versiossa voidaan mahdollisesti käyttää kyseistä palvelua, mutta toistaiseksi tämän kehitysvaiheen aikana on käytettävä vaihtoehtoisia ratkaisuja. Yksi ratkaisu tähän ongelmaan on selvittää palvelintietokoneen IP-osoite aina ennen yhteyden muodostamista. IP-osoite voidaan tarkistaa suoraan palvelimelta ja ohjelmoida se asiakaspuolen ohjelmaan aina ennen TCP-yhteyden muodostamista, mutta se ei ole tyydyttävä ratkaisu pitkällä aikavälillä. Toinen ratkaisu on sijoittaa palvelinsovellus webhotelliin, jotta se olisi aina tavoitettavissa palveluntarjoajan antaman osoitteen kautta. Ongelmaksi voi kuitenkin osoittautua, että palvelinsovellusta ei voida asentaa tai käyttää webhotellin tarjoamalla laitteistolla. Esimerkiksi Node.js -pohjainen palvelinsovellus voi olla tuettu useiden webhotellien palvelimilla, mutta mahdollisuus tukea LabView-pohjaista sovellusta voi olla harvinaisempaa.

4.2.4 Palvelinsovelluksen toiminnallinen määrittely

Kommunikaatio palvelinsovelluksen kanssa toteutetaan TCP-yhteydellä. Palvelinsovelluksen tehtävä on olla aina valmiina vastaanottamaan mahdollisia yhteydenottoja. Koska yhteydenottoja voi saapua milloin tahansa, palvelinsovellus on käynnissä jatkuvasti. Palvelinsovellus vastaanottaa yhteydenoton kun mittausjärjestelmä käynnistetään. Yhteyttä ylläpidetään niin kauan kun mittausjärjestelmä on käynnissä. Yhteyden on oltava jatkuva, jotta palvelinsovellus tietää mittausjärjestelmän olevan käytettävissä.

Samalla kun palvelinsovellus odottaa ja ylläpitää yhteyksia mittausjärjestelmiltä, se odottaa yhteyksiä käyttölaitteilta. Käyttäjä voi varata jonkin palvelinsovellukseen yhdistyneen mittausjärjestelmän ja lähettää sille käskyn siirtyä mittaustilaan. Niin kauan kun mittausjärjestelmä on mittaustilassa, muut käyttäjät eivät voi varata sitä. Palvelinsovellus vastaanottaa mittausdataa ja välittää sen käyttölaitteelle. Sovellus vastaanottaa myös muita käskyjä käyttäjältä, joita voidaan verenpaineen mittauksen suorittamiseen ja mittaustilan lopettamiseen. Mittaustilasta poistuttaessa mittausjärjestelmä vapautetaan muille käyttäjille.

4.3 Käyttölaite

Terveystuoli-järjestelmä on suunniteltu käytettäväksi käyttölaitteelta. Käyttölaitteella on kaksi tehtävää: mittausdatan näyttäminen ja mittausjärjestelmää ohjaavien käskyjen lähettäminen. Käyttölaitteen ei tarvitse olla näihin tehtäviin erikoistunut laite, vaan voisi olla käytännöllisintä jos terveystuoli-järjestelmän käyttöliittymä olisi sovellus, jota voidaan suorittaa käyttäjän omalla tietokonetta tai puhelimella. Käyttöliittymänä toimiva sovellus voisi olla verkkosovellus, jolloin se olisi käytettävissä verkkoselaimen kautta. Suuri osa nykyisistä henkilökohtaisista tietokoneista ja älypuhelimista sisältää verkkoselaimen, joten sovelluksen käyttöönotto olisi helppoa. Verkkosovellus voidaan kehittää HTML-dokumenttina, joka sisältää ohjelmoituja toimintoja. Ohjelmakoodissa on syytä käyttää JavaScript-ohjelmointikieltä, koska verkkoselaimet yleensä sisältävät JavaScript-tulkin. Ohjelmakoodin toteutus jollain muulla kielellä voisi edellyttää käyttäjää asentamaan laitteelleen ylimääräisiä ohjelmia tai lisäosia, mikä aiheuttaisi ylimääräistä vaivaa.

4.3.1 Käyttöliittymässä vaaditut toiminnot

Numeerisen datan esittäminen

Suurin osa datasta lähetään mittausjärjestelmästä lukuna, joka voidaan suoraan lukea fysiologisena arvona. Mittausdata koostuu yhdestä tavusta, joka voi saada 256 eri arvoa. Tavun sisältö tulkitaan suoraan mittaussuureen arvona ilman skaalausta. Tällöin esimerkiksi verenpaine voi sijoittua välille 0–255 mmHg, yhden yksikön resoluutiolla. Normaalin verenpainealueen selkeästi ulkopuolella olevat luvut tarkoittavat virhetilanteita, eikä niitä tulisi esittää mittaustuloksina. Esimerkiksi datatavun sisällön ollessa 16 tarkoitetaan virhetilannetta, jossa mansetti on kiinnittämättä tai huonosti kiinnitetty. Vastaavasti veren happisaturaatiomittauksessa järkeviksi

mittaustuloksiksi voidaan olettaa ainoastaan lukuja, jotka ovat enimmillään 100. Mittaustulosten ollessa järkeviä, ne voidaan näyttää verkkosovelluksessa.

EKG-käyrän esittäminen

Käyttöliittymän on kyettävä näyttämään käyttäjälle lukuarvojen lisäksi EKG-käyrän. Käyrän esittäminen eroaa merkittävästi muista mittaustuloksia, koska sitä ei voida esittää yksittäisenä lukuarvona. EKG-käyrä on muodostettava keräämällä joukko peräkkäisiä EKG-mittaustuloksia ja piirtämällä ne ajan funktiona. Koska PM6750 näytteistää EKG-signaalia taajuudella 250 Hz, voidaan olettaa, että näytteiden väli on noin $\frac{1}{250}\ s = 4\ ms$. Jos mittauspisteet asetetaan käyrän x-akselille 4 ms:n välein, EKG-käyrästä voidaan määrittää aikariippuvaisia suureita kuten sykettä.

EKG-käyrä voidaan piirtää palvelinsovelluksessa ja lähettää kuvana käyttölaitteelle, tai piirtää vasta verkkosovelluksessa. Kuvien lähettäminen usein, esimerkiksi 250 Hz:n taajuudella, verkkoyhteyden yli voi olla kuormittavaa verkon suorituskyvyn kannalta. Jos kuvia sen sijaan lähetetään matalalla taajuudella, käyttäjän näkemä EKG-käyrä päivittyisi hitaasti, mikä voi huonontaa käyttäjäkokemusta. Tästä johtuen voisi olla järkevämpää toteuttaa käyrän piirtäminen vasta käyttölaitteella, jolloin verkkoyhteyden yli lähetetään kuvien sijaan mittaustuloksia, jotka liitetään käyttölaitteella piirrettyyn käyrään. Käyrän piirtämisessä voidaan käyttää JavaScript-pohjaisia ohjelmistokirjastoja, esimerkiksi Ole Laursenin kehittämää Flotia [30].

Mittausjärjestelmää ohjaavien käskyjen syöttäminen

Käyttöliittymää käytetään mittaustulosten esittämisen lisäksi mittausten aloittamiseen ja lopettamiseen. Nämä toiminnot toteutetaan ohjauskäskyillä, jotka välitetään mittausjärjestelmälle palvelinsovelluksen kautta. Käyttöliittymässä on oltava rajapinta, joka lukee syötteitä. Koska ohjauskäskyjä on vain muutamia, voi olla käyttäjän kannalta yksinkertaisinta esittää kaikki mahdolliset käskyt valikossa. Valikko koostuisi painikkeista, jotka ovat nimetty ohjauskäskyjen mukaan. Painikkeet voidaan toteuttaa verkkosovelluksessa HTML-elementteinä, joihin on sidottu JavaScript-tapahtumankäsittelijöitä. Käskyjen lähetys käsiteltäisiin asynkronisesti, jolloin painikkeiden käyttämisellä ei olisi vaikutusta verkkosovelluksen muihin toimintoihin.

4.3.2 Tiedonsiirto verkkosovelluksen ja palvelinsovelluksen välillä

Käyttöliittymän viiveen on oltava mahdollisimman pieni, koska mitatut fysiologiset suureet ovat aikariippuvaisia. Viiveellä tarkoitetaan, että viimeisin mittausjärjestelmän tuottama mittaustulos on käyttäjän nähtävissä mahdollisimman nopeasti mittaushetken jälkeen. Jos mitatun henkilön kunnossa tapahtuu muutos, sen on näyttävä myös käyttölaitteella mahdollisimman pian. Myös käyttöliittymän painikkeiden on toimittava mahdollisimman pienellä viiveellä, jotta järjestelmän käyttökokemus olisi hyvä.

Perinteisesti verkkosivu päivitetään luomalla se uudelleen palvelimella ja lataamalla se uudelleen verkkoselaimella HTTP:aa käyttäen. Tämä ei ole sovelluksen

kannalta ihanteellisin ratkaisu, koska palvelin vastaanottaa uutta dataa yli 250 Hz:n taajuudella. Uuden verkkosivun luominen ja lähettäminen tällä taajuudella olisi epäkäytännöllistä sen edellyttämän suuren tiedonsiirtonopeuden takia. Tämän välttämiseksi voidaan käyttää HTTP:n rinnalla toimivia menetelmiä, joilla verkkosivua voidaan päivittää korkealla taajuudella ilman että koko sivu on ladattava uudelleen. Tähän tarkoitukseen verkkosovelluskehittäjät ovat käyttäneet menetelmiä kuten Ajax [32], Microsoft Silverlight [33], Adobe Flash [34] ja WebSocket [35]. Näistä menetelmistä WebSocket on kiinnostavin sen mahdollistamien toimintojen vuoksi, ja koska tässä diplomityössä halutaan suosia avoimen lähdekoodin ratkaisuja.

4.3.3 HTML 5 WebSocket-protokolla

Tiedonsiirrossa palvelimen ja verkkosovelluksen välillä voi olla hyödyllisintä käyttää HTML5 WebSocket-pistokkeita. WebSocket-pistokkeet mahdollistavat kaksisuuntaisen tiedonsiirron, mikä on välttämätöntä tässä sovelluksessa. WebSocket-pistokkeet käyttävät oletuksena samaa porttia kuten HTTP (80) ja HTTPS (443). Nämä portit mahdollistavat SSL-suojauksen käytön. Lisäksi portteja 80 ja 443 käytettäessä yhteys voidaan muodostaa palomuurin tai välityspalvelimen läpi, mikä voi olla ongelma muissa sovelluksissa. [35] WebSocket-pistokkeiden käyttö soveltuu verkkosovellukseen paremmin kuin HTTP:n pyyntö-vastaus-malli, koska tällöin palvelinsovellus voi lähettää uutta dataa jatkuvasti, odottamatta pyyntöjä verkkosovellukselta.

WebSocket-yhteys aloitetaan verkosovelluksessa. Sen jälkeen kun verkkosovellus on ladattu selaimelle HTTP:n kautta, sovellus pyytää palvelinta vaihtamaan HTTP-protokollan WebSocket-protokollaksi. Jos palvelinsovellus ymmärtää pyynnön, HTTP-yhteys hajoaa ja se korvataan WebSocket-yhteydellä. Käytännössä WebSocket-yhteys voidaan muodostaa verkkosivulle sisälletyillä JavaScript-komennoilla, jotka käsitellään sivun latautumisen jälkeen. Tämän jälkeen sekä selain että palvelin voivat lähettää ja vastaanottaa dataa, kunnes toinen osapuoli katkaisee yhteyden. [35]

Yksinkertaisuuden takia WebSocket-toimintoja on syytä käyttää ohjelmakirjastojen kautta, joissa tarpeelliset toiminnot ovat yksinkertaistettu muutamiin funktioihin. Guillermo Rauchin kehittämä Socket.io-ohjelmakirjasto [36] voi soveltua tähän tarkoitukseen hyvin. Suuri osa yleisimmistä verkkoselaimista on yhteensopivia WebSocket-protokollan kanssa. Tämän työn kirjoitushetkellä Google Chromen, Mozilla Firefoxin ja Internet Explorerin uusimmat versiot olivat yhteensopivia. Kuitenkin jotkut muut selaimet tai niiden vanhemmat versiot voivat toimia huonosti, tai olla kokonaan toimimattomia WebSocket-protokollan kanssa.

4.3.4 Verkkosovelluksen toiminnallinen määrittely

Verkkosovelluksen käynnistäminen

Verkkosovellus käynnistyy automaattisesti, kun käyttäjä syöttää palvelimen osoitteen verkkoselaimen osoiteriville, tai käyttää osoitteeseen viittaavaa kirjanmerkkiä. Selain lähettää palvelimelle HTTP-pyynnön, johon palvelinsovellus vastaa lähettämällä takaisin HTML-tiedoston. Selain vastaanottaa HTML-sivun ja hakee siinä määritellyt muut verkkosovelluksen käyttämät resurssit. Verkkosovelluksen käyttöliittymä

ilmestyy käyttölaitteen ruudulle.

Verkkosivun sisältö

Verkkosovellus näyttää verkkoselaimella sivulta, joka sisältää painikkeita, joukon mittaussuureita sekä EKG-käyrän. Painikkeilla voidaan aloittaa mittaus, lopettaa mittaus ja aloittaa verenpaineen mittaus. Painikkeiden klikkaaminen aktivoi JavaScript-tapahtumia, jotka käsitellään verkkosovelluksessa lähettämällä viestejä WebSocket-yhteyden kautta palvelinsovellukselle.

Mittaustapahtuma

Kun mittaus aloitetaan klikkaamalla käyttöliittymässä näkyvää painiketta, verkkosovellus lähettää palvelinsovellukselle aloituskäskyn. Pian tämän jälkeen verkkosovellus alkaa vastaanottamaan WebSocket-yhteyden kautta viestejä, jotka sisältävät JSON-muotoista mittausdataa. JSON-olion sisältämä data tarkastetaan, ja se käsitellään datan tyypistä riippuen. Numeerinen mittausdata esitetään lukuarvona vastaavan mittaussuureen yhteydessä verkkosivulla. EKG-data käsitellään Flot-ohjelmakirjaston toimintojen avulla, ja siitä muodostetaan EKG-käyrä. EKG-käyrää päivitetään verkkosivulla säännöllisesti piirtämällä se uudelleen, kun uutta EKG-dataa on saapunut WebSocket-yhteyden kautta. Mittausdataa vastaanotetaan jatkuvasti, kunnes käyttäjä lopettaa mittauksen.

Verenpainemittaus

Verenpaine voidaan mitata klikkaamalla sitä vastaavaa painiketta käyttöliittymässä. Painike aktivoi JavaScript-tapahtuman, joka kehottaa palvelinsovellusta aloittamaan verenpainemittauksen. Verenpainemittaus kestää useita sekunteja, minkä jälkeen verenpainearvot päivittyvät käyttölaitteen ruudulle.

Mittaustapahtuman lopettaminen

Mittaustapahtuma lopetetaan klikkaamalla sitä vastaavaa painiketta käyttöliittymässä. Painike aktivoi JavaScript-tapahtuman, jossa palvelinsovellukselle lähetetään viesti mittauksen lopettamisesta. Tämän jälkeen verkkosovellus ei enää vastaanota mittausdataa palvelinsovellukselta. Viimeisimmät mittaustulokset pysyvät käyttölaitteen ruudulla, jotta käyttäjä voi tarkastella niitä mittausten suorittamisen jälkeenkin.

5 Terveystuoli-järjestelmän prototyypin toteutus

Tässä luvussa esitellään prototyyppi luvussa 4 määritellystä Terveystuoli-järjestelmästä. Prototyypin tarkoituksena on testata Terveystuoli-järjestelmän teknistä toteutettavuutta, osoittaa mahdollisia suunnitteluvirheitä konseptissa ja toimia alustana sen jatkokehitykselle. Se toteutetaan käyttökelpoisia toimintoja sisältävänä mallina, jotta sitä voitaisiin jatkossa käyttää testaustarkoituksiin ja palautteen keräämiseen käyttäjiltä. Prototyypin laajuuteen kuuluu osa, muttei kaikkia, lopullisen Terveystuoli-järjestelmän ominaisuuksista. Prototyypin onnistuminen riippuu siitä, kuinka onnistuneesti nämä toiminnot toteutettiin. Tavoitellut toiminnot ovat listattuna:

- mittaaminen
- mittausdatan välittäminen mittausjärjestelmästä käyttölaitteelle
- mittaustulosten esittäminen
- langaton toimivuus

5.1 Mittausjärjestelmän toteutus

Terveystuoli-konseptin kannalta on tärkeää, että mittausjärjestelmä kykenee mittaamaan useita suureita samanaikaisesti ja että mittaukset ovat monipuolisia. Näiden ominaisuuksien demonstroimiseksi erilaisten mittausten joukosta valittiin muutama olennaisin. Verenpainemittausta voidaan pitää olennaisena, koska sen käyttö edellyttää kaksisuuntaista tiedonsiirtoa koko järjestelmän kautta. EKG-mittauksen toteutusta voidaan myös pitää olennaisena, koska sen tuottama data on muista Terveystuoli-järjestelmän mittauksista poiketen aaltomuotoista. Muut mittaukset tuottavat numeerisia mittaustuloksia, joiden käsittely ja esittäminen on huomattavasti yksinkertaisempaa. SpO2-mittaus sisällytetään prototyyppiin edustamaan lukuarvoina esitettäviä jatkuvia mittauksia. EKG:ta, verenpainetta ja SpO2:ta vastaavat signaalit voidaan kerätä luvussa 4.1 esitellyllä PM6750:lla, joka on liitettynä Arduino Yuniin.

5.1.1 Arduino Yunin 32U4-mikrokontrollerin käyttöönotto ja konfigurointi

Mikrokontrollerin (ATmega32U4) ohjelmoinnissa käytettiin avoimen lähdekoodin integroitua ohjelmointiympäristöä (IDE). Ohjelmisto ladattiin Arduinon sivuilta osoitteesta http://www.arduino.cc/en/Main/Software ja asennettiin kannettavaan tietokoneeseen. Kannettava tietokone käytti Windows 8 -käyttöjärjestelmää, joka oli yhteensopiva Arduino IDE:n kanssa. IDE sisältää toiminnon, jonka kautta ohjelmia voidaan kääntää ja ladata erilaisiin Arduino-laitteisiin USB-kaapelin kautta. IDE:n lataustoimintoa voidaan käyttää Arduino Yunissa ainoastaan 32U4-mikrokontrollerille. Tiedostojen siirtämiseen Arduino Yunin Linux-ympäristöön käytettiin eri menetelmää joka esitellään myöhemmässä kappaleessa. Mikrokontrollerin ohjelmoinnissa noudatettiin näitä askelia:

- liitä kaapeli tietokoneen ja Arduino Yunin Micro-B USB-portin välille
- käynnistä Arduino IDE
- valitse IDE:n asetuksista kehitysalustaksi Arduino Yun ja varmista että ohjelma on löytänyt USB-kaapelia vastaavan portin
- käännä ja lataa ohjelma mikrokontrollerille
- odota kunnes Arduino IDE on valmis

Tätä työtä tehdessä todettiin, että Arduino IDE tai mikrokontrolleri saattaa jumiutua jos näitä askelia ei noudateta. Vaikka mittausjärjestelmän lopullisessa versiossa ei käytetty 32U4-mikrokontrolleria, se todettiin käyttökelpoiseksi Terveystuoli-järjestelmän seuraavia versioita ajatellen, joissa mittausjärjestelmään saatetaan lisätä paino- ja bioimpedanssimittaukset.

Mittausjärjestelmän kehityksen aikana oli oleellista käyttää USB-yhteyttä mikrokontrollerin ja Arduino IDE:n väliseen kommunikaatioon myös ohjelman suorituksen aikana. IDE:ssa on Serial Monitor -niminen työkalu, jota voidaan käyttää tekstipohjaisena päätteenä USB-yhteyden yli. Pääte avataan valitsemalla Serial Monitor -toiminto Arduino IDE:ssa. Tämä työkalu muodostaa sarjaväylän tietokoneen ja mikrokontrollerin välille. Päätettä hyödynnettiin Arduino Yunin Linux-ympäristön komentorivin käyttämiseen 32U4-mikrokontrollerin kautta. Serial Monitor -työkalua ei käytetä mittausjärjestelmän lopullisessa versiossa, mutta se oli hyödyllinen järjestelmän kehityksen aikana. Serial Monitoria käytettiin muun muassa Arduino Yunin Linux-ympäristön IP-osoitteen tarkistamiseen, jonka jälkeen siihen voitiin muodostaa SSH-yhteys.

5.1.2 Linux-ympäristön konfigurointi

Mittausjärjestelmän normaalin toiminnan kannalta on oleellista, että se voi muodostaa internetyhteyden palvelinsovelluksen kanssa. Myös opkg-pakettien ja muiden sovellusten asennus järjestelmän kehityksen aikana edellytti internetyhteyttä. Tässä sovelluksessa verkkoyhteyden muodostamisessa käytetään Arduino Yunin sisäänrakennettua Wi-Fi-toimintoa. Kun Yun käynnistetään tehdasasetuksien pohjalta, se muodostaa oman langattoman verkon ja toimii yhteyspisteenä, johon voidaan liittyä toisella Wi-Fi-yhteensopivalla laitteella. Kun verkkoon liittynyt laite on saanut Yunin muodostamasta verkosta IP-osoitteen, voidaan käynnistää selain ja yhdistää Yunin sisäiseen IP-osoitteeseen (oletusarvona 192.168.240.1). Selain näyttää verkkosivun, jonka kautta Yun voidaan konfiguroida seuraavalla käynnistyskerralla liittymään toiseen verkkoon, sen sijaan että se muodostaisi oman verkon. Tämä toimenpide voidaan toistaa, jos on tarpeen vaihtaa Yunin käyttämää verkkoa.

Mittausjärjestelmä käyttää Arduino Yunin OpenWrt-Yun-ympäristöä. Linux-pohjaisena ympäristönä sen ohjelmointi ja käyttö eroaa merkittävästi mikrokontrollerista. Sen ohjelmointiin ei ole saatavilla erikoistunutta Arduino-kehitysympäristöä, kuten 32U4-mikrokontrollerille. Tässä työssä Linux-ympäristöä käytettiin komentotulkin kautta. Komentotulkkia käytettiin muodostamalla SSH-yhteys Yunin kanssa

samaan lähiverkkoon liittyneeltä tietokoneelta. SSH-yhteys muodostetaan Yunin IP-osoitteeseen lähiverkossa. Yunin IP-osoite voidaan selvittää esimerkiksi Arduino IDE:n Serial Monitor -työkalua.

Arduino Yunin Linux-ympäristön käytössä oleva haihtumaton muisti koostuu 16 MB:n sisäisestä flash-muistista, josta Linux-ympäristö varaa 9 MB. Tämä muisti on riittämätön, jos laitteelle asennetaan useita ohjelmistopaketteja tai sovelluksia. Haihtumattoman muistin määrää voidaan lisätä liittämällä Arduino Yuniin microSD-kortti. Kortti voidaan konfiguroida Linux-ympäristön uudeksi tiedostojärjestelmäksi, jolloin käytettävissä olevan haihtumattoman muistin määrä voidaan nostaa esimerkiksi useisiin gigatavuihin, riippuen kortin koosta. MicroSD-kortin konfigurointi on kuvattu osoitteessa http://www.arduino.cc/en/Tutorial/ExpandingYunDiskSpace. Muistin määrän laajentaminen oli välttämätöntä, että kaikki mittausjärjestelmän tarvitsemat ohjelmistopaketit voitiin asentaa.

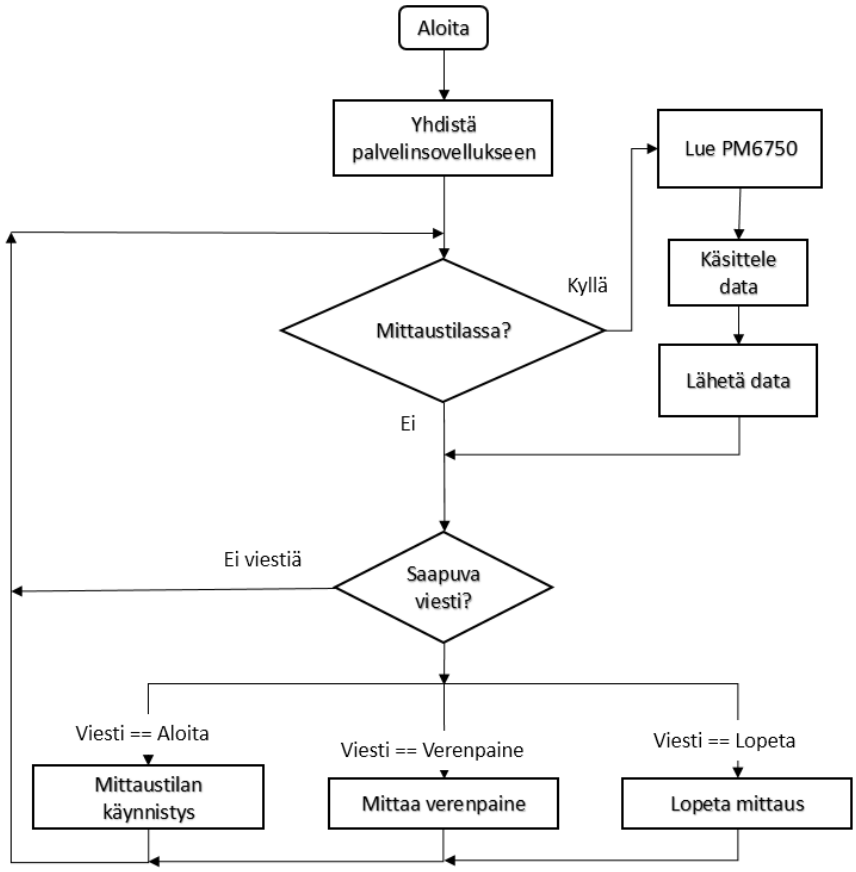
Arduino Yunin Linux-ympäristön ohjelmoinnissa käytettiin Python-skriptiä. Ohjelmointikieleksi valittiin Python, koska sitä lukeva tulkki oli oletusarvoisesti asennettu laitteelle ja ohjelman toiminnallisuuksia lisäävät moduulit olivat helposti saatavilla opkg-paketinhallintajärjestelmän kautta. Pythonin käyttö oli erityisen käytännöllistä, koska tulkittavana kielenä sitä ei tarvitse erikseen kääntää, toisin kuin esimerkiksi C-kielellä toteutettavia ohjelmia. Tässä työssä käytetty Python-skripti kirjoitettiin henkilökohtaisella tietokoneella ja ladattiin käyttöä varten Arduino Yunin Linux-ympäristön tiedostojärjestelmään SCP-tiedonsiirtoyhteydellä. Skripti voidaan myös kirjoittaa suoraan Yunilla käyttäen SSH-yhteyttä ja komentotulkkia.

PM6750 voidaan liittää Arduino Yuniin usealla eri tavalla, mutta tässä työssä käytettiin USB-sarja-muunninkaapelia (Aten UC232A). Muunninkaapelissa on USB-A ja DB9 -liittimet. Muunninkaapelin DB9-liitin liitettiin PM6750:n DB9-porttiin, joka käyttää kommunikaatiossa RS-232-sarjaprotokollaa. Muunninkaapelin konvertteri muuntaa RS-232-protokollan mukaisen datan USB-protokollan mukaiseksi. Kaapelin USB-liitin voidaan liittää suoraan Arduino Yunin USB-isäntä-porttiin. PM6750 sisältää myös USB-portin, joka voidaan kytkeä suoraan Arduino Yuniin tavallisella USB-kaapelilla, mutta silloin on käytettävä PM6750:n valmistajan tarjoamia laiteajureita jotta USB-yhteyttä voidaan käyttää. PM6750:n USB-ajuria ei löytynyt Arduino Yunin käyttöjärjestelmälle. Tämän takia käytettiin muunninkaapelia, jonka vaatima ajuri oli saatavilla myös opkg:n kautta. Aten UC232A käyttää ajuria, joka on sisällytetty opkg-pakettiin kmod-usb-serial. Ajurin asentamisen jälkeen, muunninkaapeli näkyi OpenWrt-Yunin tiedostojärjestelmässä porttina sijainnissa /dev/ttyUSB0.

5.1.3 Python-skripti

Mittausjärjestelmän ohjelmalliset toiminnot toteutettiin Python-skriptillä. Mittausjärjestelmän toiminnan ymmärtämiseksi ja mahdollisen jatkokehityksen kannalta on hyödyllistä ymmärtää miten mittausjärjestelmän Python-skripti toimii. Tässä luvussa esitellään skriptin eri toiminnallisuudet sekä kuvaillaan lyhyesti sen kehitysvaiheet. Skriptin toiminta on havainnollistettu pääpiirteissään kuvassa 12.

Python-skriptin sarjakommunikaatio toteutettiin käyttämällä pySerial-moduulia. PySerial ei kuulu Arduino Yunin tehdasasetuksiin, mutta se voitiin asentaa opkg:n



Kuva 12: Mittausjärjestelmän Python-skriptin algoritmi vuokaaviona esitettynä.

kautta pyserial-nimisenä pakettina. Asennuksen jälkeen moduuli voidaan hakea Python-skriptissä import serial -komennolla. Moduuli mahdollistaa serial-luokan käyttämisen. Luokkaa käyttäen voidaan luoda olio, jonka parametreina annetaan sarjaportin ominaisuudet, kuten portin sijainti tiedostojärjestelmässä ja baudinopeus. Sarjaportin puskurin lukemisen ja siihen kirjoittamisen mahdollistavat metodit ovat sisällytetty serial-luokkaan. Puskurista luettaessa on määriteltävä kuinka monta tavua luetaan. Jos puskurista yritetään lukea enemmän tavuja kuin mitä se sisältää, metodi jää odottamaan kunnes riittävä määrä dataa on vastaanotettu, tai odottamisen aikaraja on ylitetty.

Ohjelman toiminnan pysäyttäminen datan odottamisen ajaksi ei ole tehokasta, koska ohjelma voi odottamisen sijaan käsitellä palvelimelta saapuneita komentoja. Turhan odottamisen välttämiseksi ohjelma tarkistaa puskurin pituutta kunnes tietty tavumäärä ylittyy, jonka jälkeen puskurin data käsitellään. Toisin kuin puskurin datan lukeminen, sen pituuden tarkistus ei pysäytä koko ohjelmaa, joten sen ohella voidaan käsitellä palvelimelta saapuvaa dataa. Dataa ei ole järkevää käsitellä liian lyhyinä jonoina, koska jokainen funktio- tai metodikutsu on Python-kielessä suhteellisen hidas. Hitaus johtuu siitä, että Pythonissa jokainen kutsu luo kutsupinoon uuden kehyksen, joka suoritetaan ennen aikaisempia kehyksiä. On nopeampaa käsitellä data

yhdellä kutsulla, kuin sama määrä usealla peräkkäisellä kutsulla. Toisaalta puskuri ei saisi olla liian pitkä, koska puskurin pituus voi vaikuttaa viiveeseen datapisteen mittauksen ajanhetken ja sen esittämisen ajanhetken välillä. Mittausjärjestelmän on tarkoitus vastaanottaa sarjadataa jatkuvasti, niin kauan kun se on mittaustilassa. Tämän takia sarjapuskurin tarkistus toteutettiin while-silmukassa. Puskurissa oleva data käsitellään vasta kun sen määrä on kasvanut tarpeeksi suureksi.

Datan lähettäminen Arduino Yunista PM6750:een edellyttää vähemmän ohjelmointia kuin datan vastaanottaminen. Datan lähettämiseen voidaan käyttää Pythonin serial-luokan write()-metodia. Data lähetetään PM6750:n kommunikaatioprotokollan mukaisesti heksadesimaalimuodossa. Käytännössä PM6750:een täytyy lähettää vain muutamia käskyjä koko mittaustapahtuman aikana. EKG- ja SpO2 -parametrit mitataan automaattisesti, joten niitä ei tarvitse erikseen pyytää.

EKG mitataan oletuksena viiden elektrodin kytkennällä, joka ei ole tässä sovelluksessa haluttu mittausmenetelmä. Mittausmenetelmä voidaan vaihtaa I-kytkennäksi lähettämällä PM6750:een sopiva käsky. Tämä käsky ohjelmoitiin lähetettäväksi automaattisesti aina mittausjärjestelmän käynnistyessä, ja se jää voimaan niin kauan kun mittausjärjestelmä on käynnissä. Toinen tarpeellinen käsky on NIBP:n aloituskäsky, joka on lähetettävä aina kun verenpaine halutaan mitata. Lähetettävät viestit ovat esitetty heksadesimaalimuodossaan taulukossa 1. Käskyn saatuaan PM6750 suorittaa mittauksen ja lähettää tulokset sarjaväylän kautta. Käskyä verenpainemittauksen aloittamiseen ei lähetetä PM6750:een automaattisesti, vaan ainoastaan kun Python-skripti vastaanottaa viestin palvelinsovellukselta. Tämän takia Python-skriptin on oltava jatkuvassa valmiudessa vastaanottaa viestejä. Koska viesti voi mittausjärjestelmän kannalta saapua milloin tahansa, saapuneet viestit tarkastetaan pääsilmukan jokaisen kierroksen kohdalla.

Taulukko 1: Mittausjärjestelmän Python-skriptissä käytetyt käskyt, joilla ohjataan PM6750:n toimintaa.

Käsky	Heksadesimaalimuodossa	Lähetetään kun
EKG I-kytkentä	\x55\xAA\x4\x05\x1\xF5	Skripti käynnistyy
Mittaa verenpaine	\x55\xAA\x4\x02\x1\xF8	Viesti palvelimelta

PM6750:n lähettämä mittausdata vastaanotetaan sarjaväylän kautta jatkuvana virtana tavuja. Tavuista muodostetaan paketteja, jotka rakennetaan luvussa 4.1.1 esitellyn kommunikaatioprotokollan mukaisesti. Python-skriptin tehtävä mittausdatan käsittelyssä on lukea tavut, muodostaa niistä paketteja ja tulkita pakettien sisältö. Tavujen luku toteutettiin niin, että skripti lukee sarjaväylän puskuria ja muodostaa sen sisällöstä taulukon. Taulukon alkioina käytettiin heksadesimaalimuotoisia merkkijonoa, koska PM6750:n kommunikaatioprotokollassa data on esitetty heksadesimaalimuodossa. Kun taulukko on muodostettu, siitä voidaan etsiä datapaketin aloitusmerkit \x55 ja \xAA. Kun aloitusmerkit ovat löytyneet, paketista voidaan kerätä haluttu data. Koska paketeilla ei ole keskenään erityistä järjestystä, jokaisesta paketista on erikseen tarkistettava mitä mittausta (EKG, NiBp ja niin edelleen)

se edustaa. Kun mittauksen tyyppi on tunnistettu, siitä voidaan hakea mittaustulokset numeerisena arvona. Kun taulukko on käsitelty, skripti jatkaa sarjapuskurin lukemista ja muodostaa uuden taulukon.

Paketin alkumerkkien etsiminen toteutettiin niin, että indeksoitu muuttuja i käy silmukassa läpi taulukon alkioita, kunnes ehdot (2) ja (3) toteutuvat samanaikaisesti

$$[i] = \backslash x55 \tag{2}$$

$$[i+1] = \backslash xAA \tag{3}$$

Mittauksen tyyppi tarkistetaan lukemalla merkkijonon alkio $[i+3]$. Mittausdata sijaitsee alkioissa

$$[i+x_1], [i+x_2], ..., [i+x_n] \tag{4}$$

jossa data-alkion indeksi x_n ja data-alkioiden lukumäärä n riippuvat mittauksen tyypistä. Esimerkiksi SpO2-mittauksessa $n = 3$, jossa x_1 merkitsee happisaturaatiota, x_2 sykettä ja x_3 mittauksen tilaa. Kaikkia alkioita ei käsitellä, koska vain osa niistä sisältää tarpeellista dataa. Python-skriptissä käsitellyt alkiot ovat esitetty alla olevassa taulukossa 2. Taulukko koostuu useista paketeista, mutta se voi myös sisältää osittaisen paketin taulukon lopussa. Pakettien kadottamisen välttämiseksi Python-skriptissä jokainen taulukon lopussa sijaitseva osittainen paketti liitetään seuraavan taulukon alkuun, jolloin se muodostaa kokonaisen paketin seuraavan taulukon alkuosassa sijaitsevan osittaisen paketin kanssa.

Taulukko 2: Informaation sijoittuminen Python-skriptin käsittelemissä taulukoissa.

Tyyppi	Tunniste	1. alkio	2. alkio	3. alkio
EKG	\x01	[i+4](Aalto)		
NiBp	\x03	[i+6](Yläpaine)	[i+8](Alapaine)	[i+4](Tila)
SpO2	\x04	[i+5](Saturaatio)	[i+6](Syke)	[i+4](Tila)

Kun PM6750:sta kerätty data on muunnettu sen kommunikaatioprotokollan mukaiseen pakettimuotoon, mittausdata kerätään ja muunnetaan avain-arvo-muotoon, jossa ensimmäinen pari ilmaisee mittauksen tyypin ja seuraavat parit mittaussuureiden lukuarvot. Avain-arvo-muotoa arvioitiin käytännölliseksi muodoksi mittausdatalle, koska sitä käyttäen on yksinkertaista tunnistaa ja eritellä mittausdata järjestelmän muissa osajärjestelmissä. Avain-arvo-muodoksi valittiin JavaScript Object Notation (JSON) [37], koska se on WebSocket-protokollan tukema. Lisäksi nykyaikaiset selaimet sisältävät oletusarvoisesti JSON-jäsentimen. Python-skriptissä data muunnettiin JSON-muotoon niin, että jokainen taulukossa 2 esitetty paketti jäsennetään JSON-olioksi, joka sisältää tiedon paketin tyypistä ja sen sisältämistä mittausarvoista. Kaikki kommunikaatio mittausjärjestelmän ja palvelinsovelluksen välillä tapahtuu JSON-muodossa avain-arvo-pareina.

5.1.4 Verkkoyhteys mittausjärjestelmän ja palvelinsovelluksen välillä

Mittausjärjestelmän Python-skripti muodostaa TCP-yhteyden palvelinsovelluksen kanssa käyttäen TCP-pistokkeita. Pistokkeiden luominen on mahdollista käyttämällä Pythonin socket-moduulia, joka haetaan skriptissä import socket -komennolla. Pistoketoiminnallisuutta edustaa socket-olio, joka luodaan kutsumalla socket()-funktiota. Tämän jälkeen pistoke voidaan yhdistää haluttuun palvelimeen käyttämällä connect()-metodia, jota kutsutaan palvelimen porttinumerolla ja verkkotunnuksella tai IP-osoitteella. Tässä työssä toteutetun Python-skriptin luoma yhteys on tavuvirtamuotoinen, eli data siirtyy yksi tavu kerrallaan säilyttäen saman järjestyksen yhteyden molemmissa päissä. Vastaanottava osapuoli on vastuussa siitä, että data kerätään virrasta järkevällä tavalla.

Datan lähettämisessä ja vastaanottamisessa käytetään socket-olion send()- ja recv() -metodeja. On tärkeää huomioida että pistoketta voidaan käyttää lukkiutuvassa tai lukkiutumattomassa tilassa. Tilaa voidaan vaihtaa setblocking()-metodilla. Lukkiutuvassa tilassa pistoke odottaa aina, kunnes metodikutsu on päättynyt onnistuneesti. Send()-metodin osalta tämä tarkoittaa, että metodin kutsumisen jälkeen Python-skripti jatkaa toimintaansa vasta kun toinen osapuoli on vastaanottanut viestin. Jos toinen osapuoli on kyseisellä hetkellä kiireinen, ohjelma odottaa kunnes viesti voidaan vastaanottaa. Jos send()-metodia kutsutaan lukkiutumattomassa tilassa, pistoke ei odota toista osapuolta. Jos palvelinsovellus ei voi sillä hetkellä vastaanottaa viestiä, ohjelma kohtaa virhetilanteen. Tässä työssä havaittiin, että lukkiutumaton tila aiheutti satunnaisesti virhetilanteita, jotka johtivat aina ohjelman kaatumiseen. Virhetilanne vältettiin sillä, että pistoke asetettiin lukkiutuvaan tilaan setblocking(1)-kutsulla aina send()-metodia käytettäessä. Recv()-metodia kutsuttaessa on tärkeää käyttää lukkiutumatonta tilaa setblocking(0)-kutsulla, jotta ohjelma voi jatkaa toimintaansa, jos dataa ei sillä hetkellä ole saatavissa.

Data lähetetään ja vastaanotetaan TCP-pistokkeiden kautta peräkkäisistä tavuista koostuvina merkkijonoina. Pythonin send()-metodin parametriksi annetaan JSON-muotoinen merkkijono. TCP-pistokkeiden toiminnan takia merkkijonoja ei voida lähettää ja vastaanottaa kokonaisuuksina, vaan ne joudutaan kokoamaan uudelleen palvelinsovelluksessa. Vastaavasti palvelinsovellukselta recv()-metodilla vastaanotetut tavut kootaan kokonaisiksi viesteiksi mittausjärjestelmän Python-skriptissä. Viestit erotellaan \n-merkillä, jonka lukemisen jälkeen sitä edeltävät merkit kootaan merkkijonoksi, joka muunnetaan JSON-olioksi. JSON-olio sisältää kaksi avain-arvoparia, jotka ilmaisevat viestin tyyppiä ja sisältöä. Python-skripti voi vastaanottaa palvelinsovellukselta kolme erilaista viestiä, joita käytetään passiivisten mittauksien aloittamiseen, verenpainemittauksen suorittamiseen ja mittauksien lopettamiseen. Viestin tarkastaminen on toteutettu lukemalla JSON-olion avain-arvo-pareja.

5.1.5 Mittausjärjestelmän runko

Mittausjärjestelmän rungoksi valittiin nelijalkainen tuoli, jonka käsinojiin on kiinnitetty neljä ruostumattomasta teräksestä valmistettua kuivaelektrodia. Kaksi elektrodia ovat kiekon muotoisia, ja ne ovat kiinnitetty käsinojan päälle. Mittaustilanteessa nämä elektrodit sijaitsevat istujan molempien ranteiden alla. Toiset kaksi elektrodia

ovat pallon muotoisia ja kiinnitetty käsinojien päihin. Elektrodien sijainti on esitetty kuvassa 13 Mittaustilanteessa istujan kämmenet lepäävät näiden elektrodien päällä. Jokaiseen neljästä elektrodista on kiinnitetty ohut sähköjohto, jonka toinen pää on viety tuolin selkänojassa sijaitsevaan kytkentäliittimeen. Kytkentäliittimeen liitettiin PM6750:n neljä raajaelektrodia (merkitty RA, LA, LL, RL), niin että ranteisiin tarkoitetut EKG-elektrodit (RA, LA) ovat sähköisessä kontaktissa kiekkoelektrodien kanssa, ja jalkoihin tarkoitetut EKG-elektrodit (LL, RL) ovat sähköisessä kontaktissa palloelektrodien kanssa. EKG-elektrodien ja tuolin käsinojien elektrodien välinen resistanssi vaihteli yleismittarilla mitattuna 1–2 Ω:n välillä.

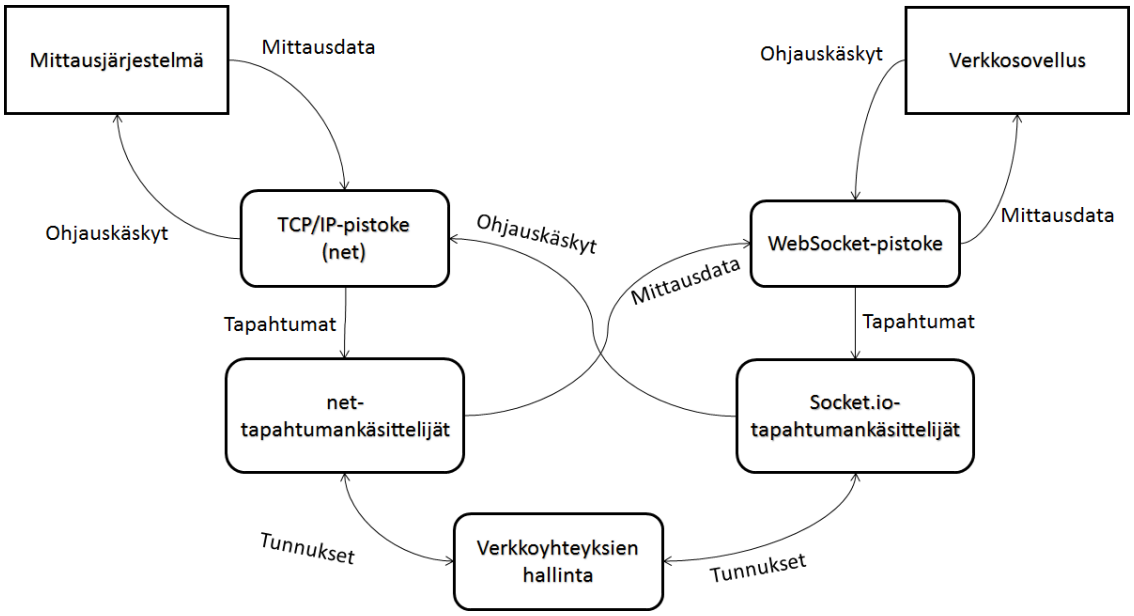
Kuva 13: Mittausjärjestelmän runkona toimiva tuoli, johon on kiinnitetty neljä elektrodia.

5.2 Palvelinsovelluksen toteutus

Tärkein tavoite palvelinsovelluksen toteutuksessa on mahdollistaa prototyypin yleinen toimivuus. TCP-toiminnallisuuksien sisällyttäminen on tärkeää, jotta sovellus voi välittää dataa mittausjärjestelmän ja käyttölaitteen välillä. TCP:n ja WebSocket-

protokollien käyttämisen lisäksi on tärkeää käyttää myös HTTP:ta, jotta verkkosovellus voidaan lähettää käyttölaitteelle ennen WebSocket-yhteyden muodostamista. Palvelinsovellus kehitettiin näiden yhteyksien mahdollistamiseksi.

Palvelinsovelluksen JavaScript-ohjelma sisältää useita asynkronisia toimintoja, joten ohjelman kuvaaminen ketjuna peräkkäisiä prosesseja on hankalaa. Sen sijaan sovellus on joukko rinnakkaisia toimintoja, joiden ajoitus riippuu yhteydenotoista sovelluksen ulkopuolelta. Palvelinsovelluksen toiminta voi olla siten helpointa havainnollistaa tietovuokaaviona, jossa ei oteta kantaa toimintojen ajoitukseen. Palvelinsovelluksen TCP-yhteyksiin liittyvät toiminnot ovat havainnollistettu alla olevassa kuvassa 14 ja kuvattu tarkemmin seuraavissa alaluvuissa.



Kuva 14: Tietovuokaavio palvelinsovelluksen toiminnasta.

Palvelinsovellus koostuu yhdestä JavaScript-tiedostosta. Sovelluksen suorittamiseksi tietokoneelle tulee olla asennettuna Node.js-ohjelmistoalusta. Palvelinsovellus käynnistetään Windows-käyttöjärjestelmällä komentorivin kautta. Koska JavaScript on tulkattava kieli, ohjelmaa ei käännetä lähdekoodista ennen sen suorittamista.

5.2.1 Yhteys mittausjärjestelmään

Palvelinsovellus käyttää samanaikaisesti kahdenlaisia TCP-yhteyksiä. Ensimmäinen näistä on toteutettu käyttäen Node.js:n net-moduulia ja on vastuussa palvelinsovelluksen ja mittausjärjestelmän välisestä tiedonsiirrosta. Net-moduulilla toteutettuja pistokkeita voidaan tässä työssä kutsua net-pistokkeiksi, ja niitä hyödyntäviä olioita net-olioiksi. Net-olioilla on sama käyttötarkoitus kuin Pythonin socket-olioilla, mutta toisin kuin socket-oliot, net-oliot vastaanottavat dataa asynkronisesti tapahtumankäsittelijöiden kautta. Käytännössä tämä tarkoittaa, että data käsitellään tapahtumankäsittelijoillä. Data siirtyy mittausjärjestelmän ja palvelinsovelluksen välisessä

TCP-yhteydessä tavuvirtana, ja se vastaanotetaan merkkijonona. Dataa vastaanottaessa palvelinsovellus lukee merkkejä ja muodostaa niistä merkkijonoa. Tavuvirran luku lopetetaan merkkiin }, joka on JSON-muotoisen merkkijonon lopetusmerkki. Merkkijono jäsennetään JSON-olioksi kutsumalla JSON.parse()-metodia käyttäen merkkijonoa parametrina. Tämän jälkeen JSON-olio lähetetään verkkosovellukselle WebSocket-pistokkeen kautta.

Kun mittausjärjestelmä yrittää muodostaa TCP-yhteyden palvelinsovellukseen, porttia kuunteleva net.Server-olio lähettää connection-tapahtuman. Connection-tapahtumaa kuunteleva tapahtumankäsittelijä luo automaattisesti net-pistokkeen ja hyväksyy TCP-yhteyden. Kun yhteys on muodostettu, mittausjärjestelmä lisätään käytettävissä olevien mittausjärjestelmien joukkoon, ja se on valmis käytettäväksi verkkosovelluksen kautta. Net-pistokkeeseen sidotaan myös kaksi vapaaehtoisesti käytettävää tapahtumankäsittelijää, jotka ovat data ja end. Data-tapahtuma lähetetään aina kun pistoke vastaanottaa dataa. Data-tapahtuman kohdalla palvelinsovellus lukee vastaanotetun datan, muuntaa sen JSON-muotoon ja lähettää sen verkkosovellukselle. Datan yhteydessä verkkosovellukselle lähetään myös dataEvent-tapahtuma, jota verkkosovellus käyttää datan tunnistamiseen. End-tapahtuma lähetetään kun mittausjärjestelmän aloittama TCP-yhteys sulkeutuu. End-tapahtuma on tärkeä huomioida, jotta useat virhetilanteet voidaan välttää. Palvelinsovellus huomioi end-tapahtuman poistamalla kyseisen mittausjärjestelmän käytettävissä olevien mittausjärjestelmien joukosta.

5.2.2 Yhteys verkkosovellukseen

Palvelinsovelluksen ja verkkosovelluksen välisessä kommunikaatiossa käytetään pääasiassa WebSocket-yhteyttä. Kuitenkin ennen kuin verkkosovellus voi muodostaa WebSocket-yhteyden, sovellus on ladattava käyttölaitteelle. Tämä toteutettiin käyttämällä palvelinsovellusta HTTP-palvelimena, joka vastaa HTTP GET-pyyntöihin lähettämällä vastauksena verkkosovelluksen HTML-tiedoston. HTTP-pyyntöjen kuuntelussa käytettiin Node.js:n Express-ohjelmistokehystä.

WebSocket-yhteys muodostetaan pistokkeilla palvelinsovelluksen ja verkkoselaimen välille, ja yhteys on jatkuva kunnes se suljetaan. Yhteyden molemmat osapuolet voivat lähettää dataa milloin tahansa. Tässä työssä WebSocket-yhteyksiä käytettiin Socket.IO-ohjelmistokirjaston kautta. WebSocket-yhteyden etu on, että sen kautta voidaan siirtää merkkien ja merkkijonojen lisäksi JSON-olioita. JSON-olioita käytetään merkkijonojen sijaan, koska JSON-olioita on helpompi käsitellä verkkosovelluksessa.

Mittausjärjestelmän toimintaa ohjaavat viestit toteutettiin palvelinsovelluksen ja verkkosovelluksen välisessä kommunikaatiossa tapahtumina, koska se oli yksinkertaisempaa kuin kommunikaation toteuttaminen esimerkiksi merkkijonoilla tai olioilla. Esimerkkeinä verkkosovelluksen lähettämistä tapahtumista voidaan mainita mittauksen aloituskäsky, lopetuskäsky ja käsky verenpainemittauksen aloittamiseen. Tapahtumien käsittelemiseksi palvelinsovelluksessa määriteltiin niitä vastaavat tapahtumakäsittelijät. Esimerkiksi aloituskäskyn tapahtumankäsittelijä käynnistää asynkronisesti funktion, jossa mittausjärjestelmän kanssa yhteydessä olevaan TCP-pistokkeeseen kirjoitetaan tietty merkkijono. Verkkosovellukselta saapuvat lopetus-

käskyt ja verenpainemittauksen aloituskäskyt toimivat vastaavalla tavalla.

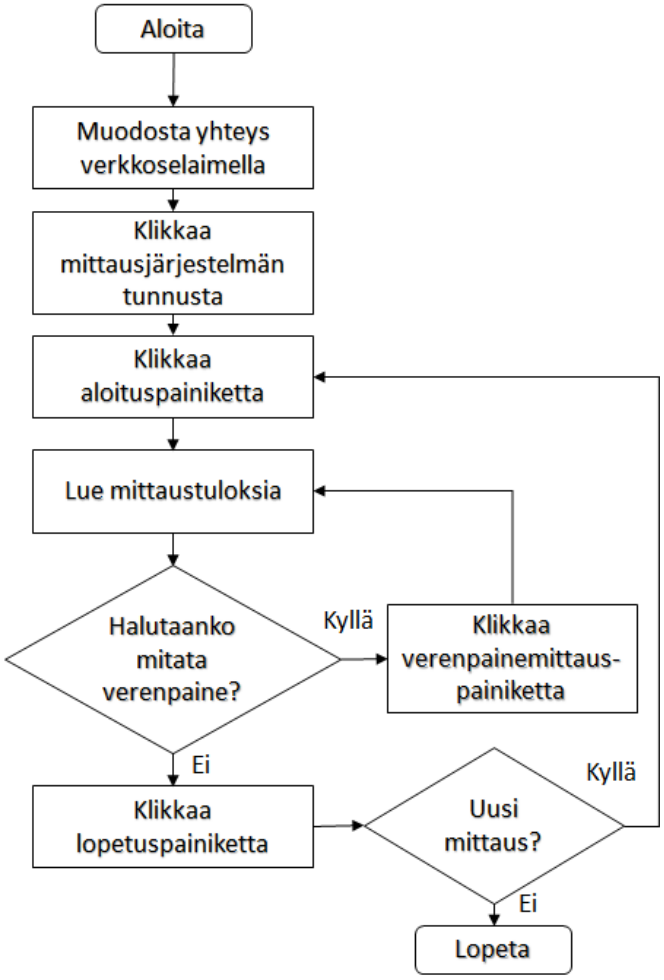
WebSocket-yhteyden etu on, että yhteyksiin voidaan sitoa valmiiden tapahtumankäsittelijöiden lisäksi itse määriteltyjä tapahtumankäsittelijöitä. Tämä mahdollistaa omien tapahtumien määrittelyn. Palvelinsovelluksessa määriteltiin neljä tapahtumaa, jotka sidottiin WebSocket-pistokkeeseen. Ensimmäinen tapahtuma on check chairs, joka lähettää verkkosovellukselle listan kaikista mittausjärjestelmistä, jotka ovat sillä hetkellä TCP-yhteydessä palvelinsovelluksen kanssa. Lista lähetetään verkkosovellukselle show chairs -tapahtuman yhteydessä, mistä verkkosovellus tunnistaa listan saapuneen.

5.2.3 Verkkoyhteyksien hallinta

Palvelinsovellukseen voi olla yhteydessä useita verkkosovelluksia ja mittausjärjestelmiä samanaikaisesti, ja sen tehtävä on välittää viestejä niiden välillä. Tämän tehtävän suorittamiseksi ohjelman on pidettävä kirjaa kaikista osapuolista. Tätä varten ylläpidetään kahta listaa, jotka koostuvat palvelinsovellukseen liittyneistä mittausjärjestelmistä ja verkkosovelluksista. Jokainen mittausjärjestelmä muodostaa TCP-yhteyden palvelinsovelluksen kanssa käynnistyessään. Niiden hallinta voidaan toteuttaa luomalla net-pistokkeista koostuva lista ja lisäämällä siihen jokainen uusi yhteys. Listan jokaisella alkiolla on oma indeksi, jota käytetään erottamaan eri mittausjärjestelmät toisistaan. Indeksi toimii mittausjärjestelmän yksilöllisenä tunnuksena. Verkkosovelluksen kautta käyttäjä voi valita listasta halutun mittausjärjestelmän. Tämän jälkeen kyseistä verkkosovellusta vastaava WebSocket-pistoke lisätään käyttäjistä muodostettuun listaan. WebSocket-pistoke sijoitetaan siihen indeksiin, jossa käyttäjän valitsema mittausjärjestelmä sijaitsee niitä vastaavassa listassa. Näin jokaisella verkkosovellus-mittausjärjestelmä-parilla on oma yksilöllinen indeksi, jonka avulla eri osapuolille tarkoitetut viestit ja mittaustulokset voidaan lähettää oikeaan kohteeseen. Kun käyttäjä lopettaa mittaamisen, hän sulkee verkkosovelluksen, ja sitä vastaava WebSocket-pistoke sulkeutuu. Tällöin se poistetaan käyttäjistä koostuvasta listasta, ja sitä vastaava mittausjärjestelmä vapautuu seuraavalle käyttäjälle.

5.3 Verkkosovelluksen toteutus

Prototyypin käyttöliittymäksi kehitettiin verkkosovellus. Verkkosovellus kehitettiin HTML-sivuna, joka haetaan palvelinsovellukselta HTTP GET-pyynnöllä. Verkkosovelluksen käyttöliittymää voidaan käyttää mittauksien aloittamiseen, lopettamiseen ja mittaustuloksien näyttämiseen. Sitä voidaan käyttää verenpaineen, veren happisaturaation, sykkeen ja EKG:n mittauksessa. Viimeksi mainittu mittaus näytetään käyttöliittymässä käyränä ja muut mittaukset lukuarvoina. Verkkosovellusta voidaan käyttää myös silloin kun Terveystuoli-järjestelmään on yhdistetty useita mittausjärjestelmiä. Jokaisella mittausjärjestelmällä on oma kokonaislukuna ilmaistu tunnus, joka näkyy verkkosovelluksessa. Tässä tapauksessa käyttäjä voi valita mitä niistä hän käyttää. Verkkosovellus on käyttäjän näkökulmasta yksinkertainen, ja sitä käytetään painikkeiden kautta. Sovelluksen toiminta käyttäjän kannalta on havainnollistettu vuokaaviona alla olevassa kuvassa 15.



Kuva 15: Verkkosovelluksen toiminta käyttäjän näkökulmasta.

5.3.1 Ohjelmointi

Verkkosovelluksen interaktiiviset toiminnot ovat toteutettu JavaScript-koodilla. Ohjelmakoodi liitettiin HTML-tiedostoon SCRIPT-elementillä. JavaScript-koodi koostuu kerran suoritettavasta osiosta, sekä JavaScript-tapahtumista, jotka suoritetaan tapahtumankäsittelijöiden kautta. JavaScript-koodi suorittaa seuraavat toiminnot yhden kerran istunnon aikana:

- WebSocket-yhteyden avaaminen palvelinsovellukseen
- vapaiden mittausjärjestelmien pyytäminen palvelinsovellukselta
- mittausjärjestelmän varaaminen

Seuraavat toiminnot ovat sidottu JavaScript-tapahtumankäsittelijöihin, ja ne voidaan suorittaa useamman kerran saman istunnon aikana:

- mittauksen aloittaminen

- mittaustulosten käsittely
- mittauksen lopettaminen

Yhteys palvelinsovellukseen

Kun HTML-sivu on ladattu ja JavaScript-koodi suoritetaan, verkkosovellus muodostaa WebSocket-yhteyden palvelinsovelluksen kanssa. WebSocket-yhteyden muodostamiseksi käytettiin Socket.io ohjelmakirjastoa, jota käytetään myös palvelinsovelluksessa. Yhteys muodostetaan luomalla pistoke-olio Socket.io-kirjaston io.connect()-metodilla. Metodi ei tarvitse palvelimen IP-osoitetta, koska yhteys rakennetaan aikaisemmin muodostetun HTTP-yhteyden päälle.

Kaikki tiedonsiirto palvelinsovelluksen ja verkkosovelluksen välillä tapahtuu WebSocket-pistokkeen metodeilla ja siihen sidotuilla tapahtumankäsittelijöillä. Dataa lähettäessä kutsutaan emit()-metodia, jolla voidaan lähettää JSON-olioita ja JavaScript-tapahtumia samanaikaisesti yhdellä metodikutsulla. JSON-olion ja tapahtuman kokonaisuutta voidaan kutsua viestiksi.

Ohjauskäskyt lähetään palvelinsovellukselle viestissä, joka sisältää JavaScript-tapahtuman ja tunnuksen, jota käytetään ohjauskäskyn välittämiseen oikealla mittausjärjestelmälle. Socket.io mahdollistaa JavaScript-tapahtumien nimeämisen, joten palvelinpuolella jokainen vastaanotettu ohjelmoitiin käsiteltäväksi eri tapahtumankäsittelijällä. Tapahtumiksi nimettiin START, jolla aloitetaan mittaustila, NIBP jolla käynnistetään verenpainemittaus ja STOP jolla lopetetaan mittaustila. Tapahtumien nimet pysyvät samoina sekä selain- että palvelinpuolella.

Ohjauskäskyjen lisäksi palvelinsovellukselle lähetetään viesti, joka sisältää CHECK CHAIRS -nimisen tapahtuman. Palvelinsovellus vastaa siihen lähettämällä listan vapaana olevista mittausjärjestelmistä. Listan perusteella käyttöjärjestelmään luodaan valikko, josta käyttäjä voi valita vapaan mittausjärjestelmän käyttöönsä.

Mittaustulokset vastaanotetaan WebSocket-yhteyden kautta viesteinä, jotka sisältävät JavaScript-tapahtuman ja JSON-olennon. JavaScript-tapahtuma nimettiin palvelinsovelluksessa DATAEVENT-nimellä. Mittaustulokset sisällytettiin mittausjärjestelmässä JSON-olentoon avain-arvo-pareina. Mittaustulosten käsittelemiseksi selainpuolella WebSocket-pistokkeeseen sidottiin tapahtumankäsittelijä, joka aktivoituu DATAEVENT-tapahtuman kohdalla. JSON-olento käsitellään tarkastamalla sen kentät. Ensimmäinen kenttä sisältää tiedon mitä mittausta JSON-olento edustaa, ja seuraavat kentät sisältävät mittaustulokset. Mittaustuloksia voi olla yksi tai useampi, riippuen mitä mittausta ne edustavat. EKG-aalto sisältää vain yhden lukuarvon, verenpaine- ja SpO2-mittaukset sisältävät kolme. JSON-olennosta poimitaan mittaustulokset, ja ne sijoitetaan oikeisiin HTML-elementteihin verkkosovelluksen käyttöliittymässä.

5.3.2 EKG-käyrän piirtäminen

EKG-käyrän piirtämisessä käytettiin Flot-ohjelmakirjastoa. Flot hyödyntää jQuery-ohjelmakirjastoa, joten molemmat niistä sisällytettiin verkkosovelluksen HTML-tiedostoon. JQueryn version 2.1.3 lähdetiedostot ladattiin osoitteesta https://

jquery.com/ ja sijoitettiin palvelinsovelluksen juurihakemistoon. HTML-sivu hakee jQuery-tiedoston palvelimelta automaattisesti GET-komennolla latautuessaan. Flot-ohjelmakirjasto haetaan internetissä olevasta tiedostovarastosta HTML-sivun latautuessa.

EKG-käyrä on ohjelmallisesti olio, joka luodaan Flot-ohjelmakirjaston metodeilla. Olio luodaan plot()-funktiolla, joka ottaa parametrina HTML-elementin ja taulukon. HTML-elementti ja sen tyylimääritykset määrittävät EKG-käyrän koon ja paikan verkkosovelluksen käyttöliittymässä. Taulukko on kaksiulotteinen ja sisältää ajan ja EKG-aaltomuodon arvon. Aikaa tarkistetaan ja ylläpidetään käyttämällä JavaScriptin Date-olioita. Date-olio sisältää päivämäärän millisekunteina, ja sitä käytetään EKG-pisteen sijoittamiseen käyrän x-akselille. Päivämäärä alustetaan olion luonnin yhteydessä, ja sitä voidaan jatkossa muuttaa esimerkiksi millisekuntimuodossa setMilliseconds()-metodilla. EKG-aaltomuodon arvoa käytetään pisteen sijoittamiseen käyrän y-akselilla.

EKG-käyrän luomisen jälkeen sitä päivitetään säännöllisesti. Käyrän päivittämiseksi on muokattava datataulukkoa, jota Flot käyttää EKG-mittauspisteiden lähteenä. Käyrä ohjelmoitiin päivitettäväksi tapahtumankäsittelijässä, joka aktivoituu aina kun verkkosovellus vastaanottaa mittaustuloksia. Kun uusi EKG-mittausarvo vastaanotetaan, Flotin käyttämästä datataulukosta ensin poistetaan vanhin alkio. Sen jälkeen verkkosovellus lisää datataulukkoon uuden alkion, joka sisältää vastaanotetun EKG-mittausarvon ja ajan. Aika määritetään lukemalla Date-olioon tallennetun edellisen ajan ja lisäämällä siihen 4 millisekuntia. 4 millisekuntia on peräisin siitä, että PM6750 näytteistää EKG:ta taajuudella 250 Hz. Kun datataulukko on päivitetty, sitä käytetään EKG-käyrän piirtämiseen uudelleen.

5.3.3 Käyttöliittymän muodostaminen HTML-elementeillä

Käyttöliittymän kehittämiseksi käytettiin HTML-elementtejä. Käyttäjälle näkyvät HTML-elementit koostuvat pääasiassa BUTTON-painikkeista ja H2-otsikoista. Painikkeita käytettiin käskyjen syöttämiseen. Otsikkoja käytettiin mittaustulosten esittämiseen. Käyttöliittymä sisältää myös EKG-käyrän, joka piirretään Flot-ohjelmakirjaston avulla.

Painikkeet, otsikot ja käyrä ovat jaettu DIV-elementteihin. Ensimmäinen DIV-elementti sisältää painikkeet, jotka ovat käyttäjälle näkyvissä heti verkkosivun latautuessa. Näitä painikkeita käytetään vapaiden mittausjärjestelmien esittämiseen käyttäjälle ja niiden valitsemiseen. Painikkeet lisätään HTML-dokumenttiin dynaamisesti JavaScript-koodilla. Painikkeet luodaan muokkaamalla olemassa olevan DIV-elementin innerHTML-ominaisuutta. Painikkeet nimetään niitä vastaavan mittausjärjestelmän yksilöllisen tunnuksen perusteella. Vapaan mittausjärjestelmän valitsemisen jälkeen painikkeet piilotetaan, ja samanaikaisesti muu osa käyttöjärjestelmästä näytetään käyttäjälle. Painikkeisiin sidottiin jQuery-ohjelmakirjaston metodeilla tapahtumankäsittelijöitä, jotka aktivoituvat painikkeita klikatessa. Viestien lähettäminen WebSocket-pistokkeen kautta palvelinsovellukselle toteutettiin näissä tapahtumankäsittelijöissä. HTML-elementtien piilottaminen ja näyttäminen toteutettiin käyttämällä DISPLAY-ominaisuutta, joka asetetaan elementin CSS-

tyylimäärittelyssä.

5.4 Prototyypin testaus

Prototyypin tarkoituksena oli sisältää neljä toimintoa, jotka esitettiin luvussa 5. Toiminnot olivat mittaaminen, mittausdatan välittäminen, mittaustulosten esittäminen ja langaton toimivuus. Prototyypin testauksella on tarkoitus selvittää, kuinka onnistuneesti nämä toiminnot toteutettiin. Kysymys toimintojen toimivuudesta ja onnistumisesta on vastattavissa laadullisella arvioinnilla. Toimintojen onnistumisen arvioinnissa voidaan käyttää havaintojen ja päätelmien lisäksi prototyyppiin liittyviä ominaisuuksia ja parametreja, kuten mittausviiveitä tai verkkoyhteyden tiedonsiirtonopeutta.

5.4.1 Metodologia

Prototyypin arvioimiseksi käytettiin tutkivaa testausta. Tutkivalla testauksella (engl. exploratory testing) tarkoitetaan tässä diplomityössä James Bachin määrittelemää (1999, [38]) menettelyä. Se sisältää joukon yksittäisiä testejä, joita käytetään tuotteen toimintojen tutkimiseen. Yksittäisiä testejä ei määritellä ennalta, vaan niitä suunnitellaan ja toteutetaan samanaikaisesti, kun edellisten testien perusteella saadaan uutta tietoa. Tämä menettely edellyttää vapauden antamista testaajalle ja vaatii vastuullisuutta testaajalta. Bachin mukaan tämä menettely on paras keino testata tuote nopeasti silloin, kun tuote kehitetään alusta asti [38].

Tutkivassa testauksessa toiminnot ovat keskeisiä. Toiminnoilla tarkoitetaan asioita, joita tuote tekee, ja niihin voi esimerkiksi kuulua tulosten esittely tai datan muuntaminen. Toiminnot luokitellaan ensisijaisiksi toiminnoiksi tai avustaviksi toiminnoiksi. Toiminto on ensisijainen, jos se on oleellinen tuotteen tarkoituksen kannalta. Avustava toiminto voi olla toiminto, jonka käyttäjä olettaa kuuluvan tuotteeseen, mutta ei ole sen tarkoituksen kannalta tärkeä.

Tutkivan testaamisen tarkoitus on löytää perusteita sille, miksi tuotetta ei voitaisi pitää toimivana. Tuote voi joko läpäistä testin, tai joutua hylätyksi. Hyväksytystä tuotteesta ei välttämättä näy, että se toimii "oikein." Mutta jos tuotteesta voidaan nähdä, että se *ei* toimi oikein, tuote hylätään. Bachin määrittelemistä läpäisykriteereistä seuraavia voidaan soveltaa tässä työssä:

Läpäisy

1. ensisijaiset toiminnot näyttävät toimivan takoituksenmukaisesti, riippumatta tuloksen oikeellisuudesta
2. tuote ei sisällä havaittavia virheitä, joka haittaavat vakavasti tuotteen normaalia käyttöä
3. tuotteen ei havaita jumittuvan, kaatuvan tai kadottavan dataa
4. yhdenkään ensisijaisen toiminnon ei havaita vikaantuvan tai muuttuvan toimimattomaksi testauksen aikana

Hylkäys

1. ainakin yksi ensisijainen toiminto näyttää toimivan väärin
2. tuote sisältää havaittavan virheen, joka haittaa vakavasti sen normaalia käyttöä
3. tuotteen havaitaan jumittuvan, kaatuvan tai kadottavan dataa
4. ainakin yhden ensisijaisen toiminnon havaitaan vikaantuvan tai muuttuvan toimimattomaksi testauksen aikana

Kriteereitä 1 ja 2 arvioitaessa otetaan huomioon mihin tarkoitukseen tarkasteltavaa toimintoa käytetään ja mitä voidaan pitää normaalina käyttönä. Kriteereitä 3 ja 4 testatessa halutaan tarkoituksellisesti aiheuttaa mahdollisia ongelmatilanteita, esimerkiksi syöttämällä vääränlaista dataa, tai käyttämällä toimintoja väärin. Testauksen aikana pyritään myös tunnistamaan ja kirjaamaan parametrejä, jotka kuvaavat tuotteen toimintakykyä.

Prototyypin testauksessa määriteltiin ensin tuotteen tarkoitus ja sen ensisijaiset toiminnot. Testaaja suunnitteli ja suoritti erilaisia testejä, kunnes kaikki mahdollisena pidetyt ongelmatilanteet käytiin läpi. Kun ongelma havaittiin, se yritettiin korjata. Tämän jälkeen testi toistettiin. Virheitä etsittiin ja korjattiin, kunnes niitä ei enää löydetty. Tärkeinä pidetyt havainnot kirjattiin. Prototyypin tarkoitus ja toiminnot määriteltiin seuraavasti:

Tarkoitus

– fysiologisten suureiden mittaaminen ja mittaustulosten esittäminen

Ensisijaiset toiminnot

– mittaaminen
– mittausdatan välittäminen mittausjärjestelmästä käyttölaitteelle
– mittaustulosten esittäminen
– langaton toimivuus

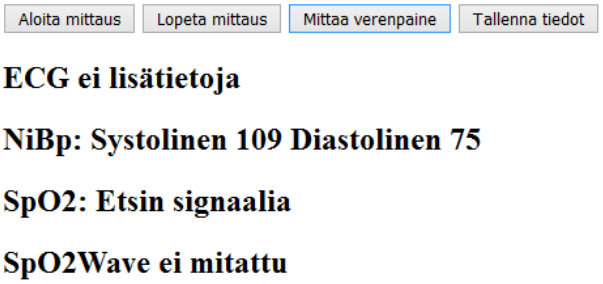
5.4.2 Tulokset

Mittaaminen

Prototyyppi tuottaa mittausdataa PM6750:n kautta. Mittausdatan tuottamista testattiin aluksi yksikkötesteillä, joista edettiin vähitellen laajempiin integrointitesteihin. Yksinkertaisin testi toteutettiin muodostamalla SSH-yhteys ulkoiselta tietokoneelta Arduino Yuniin ja käynnistämällä komentorivin kautta Python-skripti, joka tulosti tietokoneen ruudulle PM6750:n lähettämää mittausdataa. Tuloste oli heksadesimaalimuotoista, joten sen ymmärtäminen oli hankalaa. Python-skriptin kehityksen edetessä mittausdata voitiin esittää desimaalimuotoisena, jolloin mitatut suureet

voitiin helposti lukea. Python-skriptin lopullisessa versiossa tulokset voitiin lähettää palvelinsovellukselle, jolloin mittausjärjestelmän voitiin todeta toimivan tarkoituksenmukaisella tavalla. Kun mittaukset havaittiin toimiviksi, niihin liittyviä viiveitä mitattiin sekuntikellolla.

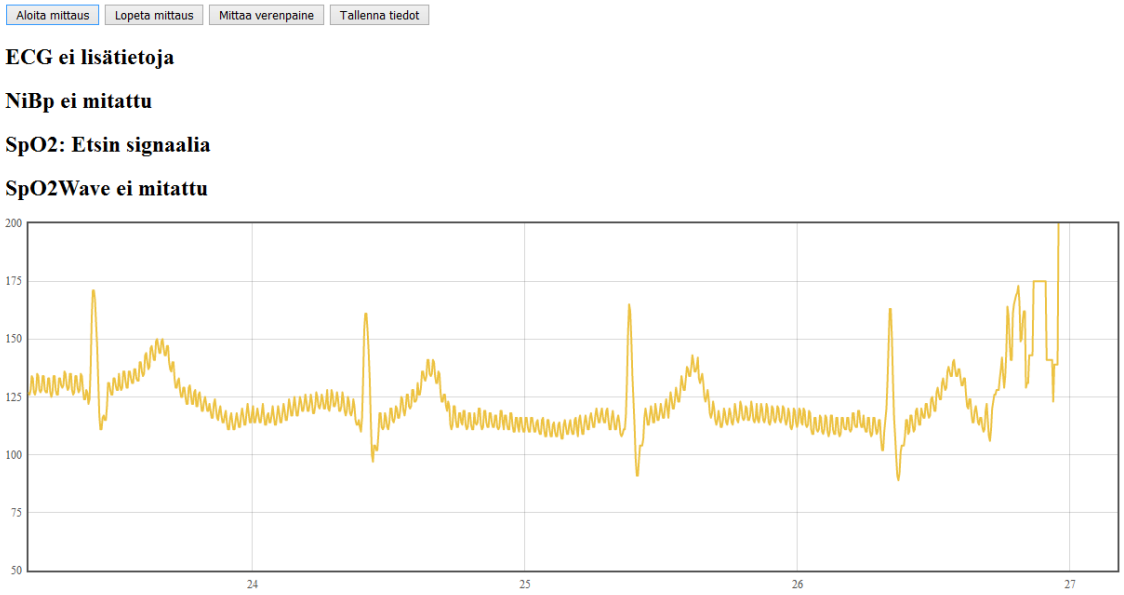
Verenpaineen mittaus oli toimiva. Ennen mittauksen testaamista PM6750:n mansetti puettiin olkavarren ympärille ja kiristettiin. Mittaus aloitettiin painamalla käyttöliittymässä näkyvää painiketta. Jos mansetti oli huonosti kiinni, sen täyttäminen keskeytyi automaattisesti, ja virheilmoitus ilmestyi käyttölaitteen ruudulle. Onnistuneen mittauksen jälkeen systolisen ja diastolisen verenpaineen lukemat ilmestyivät käyttöliittymään (ks. kuva 16). Verenpainemittaukseen liittyi kaksi aikasuuretta, joista ensimmäinen oli mittauksen käynnistymisen viive ja toinen mittaukseen kulunut aika. Verenpainemittauksen käynnistymisen viive määriteltiin verenpainemittauspainikkeen painamisen ja verenpainemittauksen aloittamisen välillä kuluvaksi ajaksi. Verenpainemittaus katsottiin alkaneeksi silloin, kun PM6750 alkoi täyttämään verenpainemansetin ilmapussia. Painikkeen painaminen ja viiveen havainnointi toistettiin kymmenen kertaa, ja viive oli toistuvasti alle 500 ms. Tätä viivettä voidaan pitää riittävän pienenä, jotta siitä ei ole häiriötä käyttäjälle. Verenpainemittaukseen kulunut aika arvioitiin kymmenen toiston testissä ja se vaihteli noin 30 sekunnin ja 40 sekunnin välillä. Tämä aikaväli määriteltiin alkaneeksi siitä hetkestä, kun verenpainemansetin ilmapussin täyttäminen alkoi ja päättyväksi siihen hetkeen, kun mittaustulokset ilmestyivät verkkosovelluksen käyttöliittymään.



Kuva 16: Prototyypillä suoritetun verenpainemittauksen tulokset käyttöliittymässä esitettynä.

EKG:n mittaus oli toimiva. Sen testaamiseksi henkilö istuutui mittausjärjestelmän tuolille, asetti ranteensa sylinterimäisten elektrodien päälle ja kätensä pallomaisten elektrodien ympärille. Kosketuksen jälkeen EKG:ssa näkyi äkillisiä muutoksia. Transientin jälkeen EKG-käyrä asettui tilaan, josta saattoi nähdä tunnusomaisia muotoja kuten P-aallon, QRS-kompleksin ja T-aallon. Yksittäinen EKG-näyte voi saavuttaa 250 diskreettiä arvoa välillä 1–250, joista arvot välillä 50–200 piirrettiin näkyviin. Käyrän y-akselin asteikon resoluutio todettiin riittäväksi eri aaltojen erottamiseksi toisistaan. X-akselin aika-asteikkoa voitiin käyttää sykkeen selvittämiseen käyrän jaksollisisten muotojen perusteella. Kun käyrä ajautui transientin vaikutuksesta asteikon ulkopuolelle, se palautui automaattisesti asteikon puoleenväliin, arvon 125 lähistöön. Käsien liikuttaminen mittauksen aikana näkyi häiriöinä EKG:ssa. EKG:n mittauksessa viive oli toistuvasti alle 500 ms. Viive määriteltiin aikaerona

iho-elektrodikontaktin muodostumisen ja käyttöliittymässä näkyvän vasteen välillä. Tätä viivettä voidaan pitää riittävän pienenä, jotta siitä ei ole häiriötä käyttäjälle. Alle 500 ms:n viive havaittiin myös, kun kädet nostettiin pois elektrodien päältä. Esimerkki testauksen aikana tuotetusta EKG-käyrästä on esitetty kuvassa 17. Käyrän loppuosassa näkyvä häiriö tapahtui, kun mittauksen kohteena olevan henkilön kädet nostettiin elektrodien päältä.

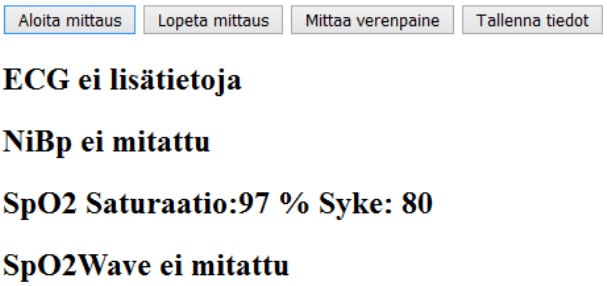


Kuva 17: Prototyypillä mitattu EKG-käyrä käyttöliittymässä esitettynä.

SpO2-mittaus oli toimiva. Sitä testattiin kiinnittämällä PM6750:n pulssioksimetri henkilön sormenpäähän ja lukemalla saturaatiotuloksen ja sykkeen käyttöliittymästä (ks. kuva 18). SpO2-mittaukseen liittyi kaksi viivettä. Ensimmäinen viive määriteltiin hetkien välille, jolloin pulssioksimetri asetettiin sormenpäähän ja jolloin happisaturaatio ja syke ilmestyivät käyttöliittymään. Viiveeksi arvioitiin kymmenen mittauskerran aikana toistuvasti noin 8–9 s. Toinen viive havaittiin, kun pulssioksimetri poistettiin sormenpäästä. Tällöin edelliset mittaustulokset näkyivät noin sekunnin ajan käyttöliittymässä, ennen kuin ne nollautuivat. Tämä viive on pieni, ottaen huomioon että SpO2-tuloksien päivitystaajuus on 1 Hz. Ensimmäinen 8–9 s:n viive johtuu PM6750:n toiminnasta, eikä sitä voitu vähentää.

Mittaustulosten esittäminen

Mittaustulosten luku edellyttää jokaisen osajärjestelmän toimintaa ja yhteistyötä. Palvelinsovelluksen ja käyttölaitteen yhteensopivuutta tutkittiin lataamalla verkkosovellus käyttölaitteelle ja testaamalla verkkosovelluksen käyttöliittymää. Verkkosovelluksen lataamiseen käytettiin älypuhelinta, kannettavaa tietokonetta ja kahta tablettitietokonetta. Testatut laitteet ovat esitetty taulukossa 3 Kaikki testauksessa käytetyt laitteet onnistuivat HTTP-pyynnön lähettämisessä ja HTML-tiedoston vastaanottamisessa palvelimelta, jolloin verkkosovelluksen käyttöliittymä ilmestyi



Kuva 18: Prototyypillä suoritetun happisaturaatiomittauksen tulokset käyttöliittymässä esitettynä.

laitteen ruudulle. Tämän jälkeen tutkittiin mittausjärjestelmän valitsemista, mittausten käynnistämistä ja tulosten lukemista. Testin ajan käyttölaite, palvelinsovellus ja mittausjärjestelmä olivat yhdistetty samaan lähiverkkoon. Mittausjärjestelmän valitseminen ja käyttäminen toimi normaalisti kaikilla testatuilla laitteilla, paitsi yhdellä tablettitietokoneista. Kyseisen tablettitietokoneen Android 3.2 -käyttöjärjestelmän internetselain havaittiin yhteensopimattomaksi WebSocket-protokollan kanssa. Toisessa tablettitietokoneessa oli Android 4.4.2 -käyttöjärjestelmä ja selaimena Chrome 45.0.2454.84, joka toimi normaalisti.

Taulukko 3: Testatut käyttölaitteet.

Laite	Malli	Käyttöjärjestelmä	Selain	OK
Tabletti	Lenovo TAB 2 A7-10F	Android 4.4.2	Chrome 45.0.2454.84	x
Tabletti	Samsung GT-P6200	Android 3.2	Internet 3.2-ZSLA2	
Puhelin	Samsung GT-S5570	Android 2.2.1	Internet 2.2.1	x
PC	Acer Aspire V5-573G	Windows 8	Mozilla Firefox 40.0.3	x

Mittausjärjestelmän ohjaaminen

Mittausjärjestelmän ohjaamista testattiin aluksi suoraan Arduino Yunin kautta. Python-skripti ohjelmoitiin lähettämään verenpainemittauskäsky PM6750:lle ohjelman käynnistyessä. Myöhemmissä versioissa mittauskäskyt lähetettiin palvelinsovellukselta TCP-yhteyden välityksellä. Prototyypin nykyinen versio ohjaa mittausjärjestelmää verkkosovelluksen kautta. Koska käyttäjältä kerätään syötettä vain painikkeiden kautta, järjestelmää ei ole normaalissa käytössä mahdollista jumittaa tai kaataa manipuloimalla syötettä.

Mittausdatan välittäminen mittausjärjestelmästä käyttölaitteelle

Prototyypin palvelinsovellusta ajettiin kannettavalla tietokoneella, joka oli yhdistetty mittausjärjestelmän ja käyttölaitteen kanssa samaan WLAN-verkkoon. Palvelinsovel-

lus kykeni muodostamaan ja katkaisemaan yhteyksiä saman istunnon aikana niin usein kuin oli tarpeellista. Mittausjärjestelmän Python-skripti muodosti TCP-yhteyden palvelinsovelluksen kanssa ilman ongelmia, kun langaton lähiverkkoyhteys oli käytettävissä. Järjestelmän havaittiin toimivan vakaasti, jos mittausjärjestelmästä tai käyttölaitteesta katkaistiin äkillisesti virta mittauksen aikana. Useita TCP-yhteyteen liittyviä vikaantumismoodeja havaitiin ja korjattiin.

Terveystuoli-järjestelmän yleisen käytettävyyden kannalta olisi suositeltavaa, että palvelinsovellus on käytettävissä verkkotunnuksen, alitunnuksen tai muun pysyvän osoitteen kautta. Tilapäisen IP-osoitteen käyttämistä ei voida pitää tyydyttävänä ratkaisuna, koska tällöin mittausjärjestelmän Python-skriptiä on aina muokattava palvelimen IP-osoitteen vaihtumisen jälkeen. Osoitteen vaihtuminen on haitaksi myös käyttäjälle, koska sen selvittäminen aiheuttaa ylimääräistä vaivaa. Siksi suositellaan, että palvelinsovelluksen pitkäkestoiseksi ylläpitämiseksi käytetään verkkotunnusta tai alitunnusta.

Langaton toimivuus

Prototyyppiä testattiin kaikkine osajärjestelmineen ympäristössä, jossa Wi-Fi-signaalin laatu oli heikko. Mittausjärjestelmä, palvelinsovellus sekä käyttölaite liitettiin langattomaan lähiverkkoon Wi-Fi-yhteyden avulla. Samassa ympäristössä käytetettynä heikoin signaali-kohina-suhde havaitiin mittausjärjestelmässä, joka käyttää Arduino Yunin Wi-Fi-lähetin-vastaanotinta. Signaali-kohina-suhde todettiin huonoksi, kun se oli 5–15 dB tai pienempi. Tällöin havaittiin, että mittausjärjestelmän ja palvelinsovelluksen välinen verkkoyhteys hidastui ja katkeili ajoittain, mikä häiritsi datan lähettämistä ja vastaanottamista. Näiden ongelmien lähteen todentamisessa käytettiin Pythonin virheilmoituksia. Virheilmoitukset paikannettiin Python-skriptin pistoke-olioiden toimintaan, tarkemmin sanottuna send()- ja recv()-metodeihin. Kyseisiä ongelmia ei havaittu toisessa ympäristössä, jossa Wi-Fi-signaalin signaali-kohina-suhde oli suurempi kuin 20 dB. Siksi on suositeltavaa että Wi-Fi-yhteyttä käytettäessä huomioidaan ympäristön radiotaajuiset häiriöt ja WLAN-tukiaseman sijainti suhteessa mittausjärjestelmään.

Prototyypin osajärjestelmien välisiä tiedonsiirtonopeuksia tutkittiin palvelimelta käsin. Tässä käytettiin Windows-käyttöjärjestelmän Tehtävienhallinta-työkalua, joka näyttää millä prosesseilla on verkkotoimintaa. Mittaustapahtuman aikana palvelinsovelluksen havaittiin vastaanottavan mittausjärjestelmältä keskimäärin 7240 tavua sekunnissa ja lähettävän verkkosovellukselle 11860 tavua sekunnissa. Nämä luvut ovat yhtä mittausjärjestelmää ja yhtä käyttäjää kohden, ja samanaikaisten mittausten lukumäärää lisäämällä palvelinsovelluksen vastaanottama ja lähettämä data lisääntyy vastaavasti.

5.4.3 Tulosten tulkinta

Edellä kuvailtuja toimintoja arvioitiin luvussa 5.4.1 esitettyjen kriteereiden mukaan. Ensisijaiset toiminnot eli mittaaminen, mittausdatan välittäminen, tulosten esittäminen ja langattomat toiminnot havaittiin toimivan tarkoituksenmukaisesti. Näihin

toimintoihin liittyvät parametrit koottiin alla olevaan taulukkoon 4. Ongelmia havaittiin, kun langattoman verkon Wi-Fi-signaalin laatu oli heikko ja kun käyttölaitteen verkkoselain ei ollut yhteensopiva verkkosovelluksen käyttämän tekniikan kanssa. Nämä tilanteet kuitenkin sijoittuvat normaalin käytön ulkopuolelle. Normaalia käyttöä haittaavia virheitä ei havaittu. Prototyyppi ei jumittunut, kaatunut tai kadottanut dataa. Prototyyppi läpäisi testausmenettelyssä asetetut kriteerit, joten sitä voidaan pitää toimivana ja onnistuneena.

Taulukko 4: Prototyypin testauksessa selvitetyt tekniset parametrit.

Suure	Arvo	Yksikkö
Mittausjärjestelmän käynnistysnopeus	80–85	s
NiBp: viive	< 500	ms
NiBp: mittaukseen kuluva aika	30–40	s
EKG: viive	< 500	ms
EKG: näytteistystaajuus	250	Hz
EKG: käyrän päivitystaajuus	3,33	Hz
SpO2: käynnistysnopeus	8–9	s
SpO2: lukeman päivitystaajuus	1	Hz
Tavunopeus: Palvelinsovellus-verkkosovellus	11,86	kB/s
Tavunopeus: Mittausjärjestelmä-palvelinsovellus	7,24	kB/s

6 Johtopäätökset ja suositukset

6.1 Johtopäätökset

Terveystuoli-järjestelmällä voidaan mitata helposti ja samanaikaisesti useaa suuretta, jotka kuvaavat potilaan terveydentilaa. Mittaustulokset ovat luettavissa mittauksen aikana lähes reaaliaikaisesti ja vähäisellä viiveellä. Kaikki mittauslaitteet järjestelmässä ovat yhdistetty keskitettyyn palveluun, jota voidaan käyttää mittaustulosten tallentamiseksi ja järjestelemiseksi. Potilasmäärän kasvaessa mittauskapasiteettia voidaan lisätä. Järjestelmän eri osilla kerätyt tiedot voidaan koota samaan tietokantaan, jolloin aikaisemmin kerättyjä tietoja ei tarvitse jälkikäteen etsiä erillisten laitteiden joukosta. Koska järjestelmän osat ovat yhteydessä internet-infrastruktuurin kautta, potilas ja käyttäjä voivat sijaita kaukana toisistaan. Esimerkiksi mittausjärjestelmän sijoittaminen potilaan kotiin on mahdollista, jolloin potilaan ei tarvitse matkustaa hoitajan luokse, tai hoitajan potilaan luokse. Hoitaja ja potilas voivat kommunikoida etämittauksen aikana kuvayhteydellä, puhelimitse, tai esimerkiksi *Voice over IP* (VOIP) -menetelmien välityksellä. VOIP-toiminnallisuuden lisääminen Terveystuoli-järjestelmään on yksi sen lukuisista kehitysmahdollisuuksista. Mahdollisuus käyttää Terveystuoli-järjestelmää verkon yli ei kuitenkaan haittaa sen käyttöä tavanomaisemmissa hoitotilanteissa.

Mittaustulosten lukemiseen kelpaa suurin osa nykyaikaisista verkkoselaimista, ja niitä voidaan käyttää erilaisilla laitteilla pöytätietokoneista älypuhelimiin ja taulutietokoneisiin. Yhteensopimaton verkkoselain voidaan useimmissa tapauksissa päivittää uusimpaan versioon, joka on yhteensopiva. Terveystuoli-järjestelmää varten ei tarvitse suunnitella tai valmistaa erillistä käyttölaitetta, mikä mahdollistaa matalammat valmistuskustannukset ja edullisemman hinnoittelun.

Terveystuoli-järjestelmän osajärjestelmät suunniteltiin mahdollisimman modulaarisiksi, eli monia toimintoja voidaan lisätä tai poistaa ilman, että koko järjestelmän toiminta häiriintyy. Nykyinen prototyyppi sisältää kolme eri mittausta, mutta mittausten lukumäärällä tai tyypillä ei ole olennaista vaikutusta palvelinsovelluksen kannalta. Muiden suureiden mittaamiseksi mittausjärjestelmään voidaan yhdistää uusia mittauslaitteita, joiden tuottama data muunnetaan sopivaan muotoon ja lähetetään TCP-yhteyden kautta palvelinsovellukselle. Terveystuoli-järjestelmää voidaan käyttää vaikka mittausjärjestelmä puuttuisi, jolloin mittaustuloksia voidaan testi- tai demonstraatiotarkoituksessa simuloida.

6.2 Suositukset

Jatkokehitystä ajatellen voidaan tunnistaa kaksi kehityskohdetta, jotka voivat lisätä Terveystuoli-järjestelmän hyödyllisyyttä fysiologisten parametrien mittaamisessa ja tulosten esittämisessä. Ensimmäinen näistä on paino- ja bioimpedanssimittausten toteuttaminen. Tämä voidaan toteuttaa liittämällä AFE4300EVM-PDK Arduino Yunin 32U4-mikrokontrolleriin ja käyttämällä mikrokontrolleria laitteen ohjaamiseen ja mittausdatan keräämiseen. On kehitettävä menetelmä lähettää mikrokontrollerilla kerätty mittausdata verkkoyhteyden kautta palvelinsovellukselle. Mittausda-

tan käsittelyssä palvelin- ja verkkosovelluksessa voidaan käyttää jo olemassa olevia JavaScript-pohjaisia menetelmiä.

Toinen mahdollinen kehittämiskohde on tietokannan lisääminen. Käyttäjätilit ja mittausten tallennus olivat osa Terttu-terveystuolia, ja lisäksi siltä toivottiin yhteensopivuutta potilastietojärjestelmän kanssa [20]. Sama toiminnallisuus olisi oletettavasti hyödyllinen tässäkin sovelluksessa. Keskitetty tietokanta voi olla yksinkertaista toteuttaa Terveystuoli-järjestelmän palvelinsovelluksessa, koska kaikki osajärjestelmien välinen data siirtyy sen läpi. Terveystuoli-järjestelmä toimii jo valmiiksi internetin välityksellä, joten on myös mahdollista liittää se johonkin olemassa olevaan potilastietojärjestelmään.

Viitteet

[1] Linnavuo, M. Projektitutkija. Aalto-yliopisto. Otakaari 7, 02150 Espoo. Haastattelu, 5.2.2015.

[2] Mustonen, V. Development of a health chair for non-invasive patient monitoring. Diplomityö, Aalto-yliopisto, Sähkötekniikan korkeakoulu, Espoo, 2013.

[3] Malmivuo, J. ja Plonsey, R. *Bioelectromagnetism - Principles and Applications of Bioelectric and Biomagnetic Fields.* New York, Oxford University Press, 1995.

[4] Lukaski, H., Hall, C. ja Siders, W. Validation of tetrapolar bioelectric impedance method to assess human body composition. *The American Journal of Clinical Nutrition,* 1996, vol. 64, s. 485–488.

[5] Millstein, R. A. Measuring Outcomes in Adult Weight Loss Studies That Include Diet and Physical Activity: A Systematic Review. *Journal of Nutrition and Metabolism,* verkkolehti, 2014, vol. 2014. Viitattu 31.8.2015. Saatavissa: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4262752/.

[6] Vilhunen, T. Biologisen materian sähkönjohtavuusominaisuuksien määritys. Pro gradu -tutkielma, Kuopion yliopisto, Sovelletun fysiikan laitos, Kuopio, 2000.

[7] Moyle, J. T. B. *Pulse Oximetry.* 2. painos. Lontoo, BMJ Books, 2002.

[8] Skirton, H., Chamberlain, W., Lawson, C. ja Young, E. A systematic review of variability and reliability of manual and automated blood pressure readings. *Journal of clinical nursing,* 2011, vol. 20, s. 602–614.

[9] Banner, T. E. ja Gravenstein, J. S. Comparative effects of cuff size and tightness of fit on accuracy of blood pressure measurements. *Journal of Clinical Monitoring,* 1991, vol. 7, nro 4, s. 281–284.

[10] Chobanian, A. V., Bakris, G. L., Black, H. R. et al. The Seventh Report of the Joint National Committee on Prevention, Detection, Evaluation, and Treatment of High Blood Pressure: the JNC 7 report. *JAMA,* 2003, vol. 289, nro 19, s. 2560-2571.

[11] Lu, S. H., Leasure, A. R. ja Dai, Y. T. A systematic review of body temperature variations in older people. *Journal of Clinical Nursing,* 2009, vol. 19, s. 4–16.

[12] Malanda, U. L., Welschen, L. M., Riphagen, I. I. et al. Self-monitoring of blood glucose in patients with type 2 diabetes mellitus who are not using insulin. *The Cochrane database of systematic reviews,* verkkojulkaisu, 2012. Viitattu 31.8.2015. Saatavissa: http://onlinelibrary.wiley.com/doi/10.1002/14651858.CD005060.pub3/abstract.

[13] Vashist, S. K. Continuous Glucose Monitoring Systems: A Review. *Diagnostics,* 2013, vol. 3, nro 4, s. 385–412.

[14] Gregg, E. W., Cheng, Y. J., Cadwell, B. L. et al. Secular trends in cardiovascular disease risk factors according to body mass index in US adults. *Journal of the American Medical Association,* 2005, vol. 293, nro 15, s. 1868–1874.

[15] Romero-Corral, A., Montori, V. M., Somers, V. K. et al.Association of bodyweight with total mortality and with cardiovascular events in coronary artery disease: a systematic review of cohort studies. *The Lancet,* 2006, vol. 368, nro 9536, s. 666–678.

[16] Suominen, M. H., Soini, H., Muurinen, S., Strandberg, T. ja Pitkälä, K. Ikääntyneiden ruokatottumukset, ravinnonsaanti ja ravitsemustila suomalaisissa tutkimuksissa. *Sosiaalilääketieteellinen aikakauslehti,* 2012, vol. 49, s. 170–179.

[17] Hannah, R. L. ja Reed, S. E. *Strain Gage Users' Handbook.* Lontoo, Chapman & Hall, 1992.

[18] Ragheb, T. ja Geddes, L. A. The polarization impedance of common electrode metals operated at low current density. *Annals of Biomedical Engineering,* 1991, vol. 19, s. 151–163.

[19] Blomqvist, K. H., Sepponen, R. E., Lundblom, N. ja Lundblom, J. An open-source hardware for electrical bioimpedance measurement. *Electronics Conference (BEC), 13th Biennial Baltic,* Tallinna, 3–5.10.2012, 2012, s. 199–202.

[20] Ronkainen, K., Fagerroth, K. ja Valkama, A. Terveystuolin havainnoinnin tulokset. Havainnointitutkimus, Laurea-ammattikorkeakoulu, Espoo, 2014.

[21] Anonyymi. PM6750 Multiparameter Monitor Module (Bluetooth) Tech manual. Käyttöopas, Shanghai Berry electronic tech co.,ltd, Shanghai, 2014.

[22] EIA standard RS-232-C. Interface between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange. Washington, Electronic Industries Association, 1969. 29 s.

[23] Lim, Y. G., Kim, K. K. ja Park, K. S. ECG measurement on a chair without conductive contact. *IEEE Transactions on Biomedical Engineering,* 2006, vol. 53, nro 5, s. 956–959.

[24] Anonyymi. AFE4300 Weigh Scale/Body Composition Analog Front End Performance Demonstration Kit. Verkkodokumentti. Päivitetty 1.9.2015. Viitattu 1.9.2015. Saatavissa: http://www.ti.com/tool/afe4300evm-pdk.

[25] Anonyymi. Arduino - ArduinoBoardYun. Verkkodokumentti. Päivitetty 1.9.2015. Viitattu 1.9.2015. Saatavissa: https://www.arduino.cc/en/Main/ArduinoBoardYun.

[26] Blank, A. G. *TCP/IP JumpStart: Internet Protocol Basics.* 2. painos. Alameda, SYBEX, 2000.

[27] Anonyymi. Integrated Analog Front-Endfor Weight-Scale and Body Composition Measurement. Käyttöopas, Texas Instruments, Dallas, 2014.

[28] Kunkle, J. Node.js Basics Explained. Verkkodokumentti. Päivitetty 1.9.2015. Viitattu 1.9.2015. Saatavissa: https://www.altamiracorp.com/blog/employee-posts/nodejs-basics-explained.

[29] Anonyymi. 7.2.1 Socket Objects. Verkkodokumentti. Päivitetty 8.2.2005. Viitattu 1.9.2015. Saatavissa: https://docs.python.org/2.3/lib/socket-objects.html.

[30] Anonyymi. Flot: Attractive JavaScript plotting for jQuery. Verkkodokumentti. Päivitetty 7.7.2014. Viitattu 2.9.2015. Saatavissa: http://www.flotcharts.org/.

[31] Anonyymi. Node.js. Verkkodokumentti. Päivitetty 2.9.2015. Viitattu 2.9.2015. Saatavissa: https://nodejs.org/en/.

[32] Garrett, J. J. Ajax: A New Approach to Web Applications | Adaptive Path. Verkkodokumentti. Päivitetty 2.9.2015. Viitattu 2.9.2015. Saatavissa: http://adaptivepath.org/ideas/ajax-new-approach-web-applications/.

[33] Anonyymi. Microsoft Silverlight. Verkkodokumentti. Päivitetty 2.9.2015. Viitattu 2.9.2015. Saatavissa: http://www.microsoft.com/silverlight/.

[34] Anonyymi. Adobe - Flash Player. Verkkodokumentti. Päivitetty 2.9.2015. Viitattu 2.9.2015. Saatavissa: http://www.adobe.com/fi/software/flash/about/.

[35] RFC 6455. The WebSocket Protocol. Internet Engineering Task Force (IETF), 2011, 71 s. Viitattu 2.9.2015. Saatavissa: https://tools.ietf.org/html/rfc6455.

[36] Anonyymi. Socket.IO. Päivitetty 2.9.2015. Viitattu 2.9.2015. Saatavissa: http://socket.io/.

[37] ECMA-404. The JSON Data Interchange Format. Geneve, Ecma International, 2013. 14 s.

[38] Bach, J. General Functionality and Stability Test Procedure. Verkkodokumentti. Päivitetty 24.5.2006. Viitattu 8.9.2015. Saatavissa: http://www.satisfice.com/tools/procedure.pdf.